UNIVERSIDADE FEDERAL DE SERGIPE
CAMPUS ALBERTO CARVALHO
DEPARTAMENTO DE SISTEMAS DE INFORMAÇÃO

YTHANNA DE OLIVEIRA GOMES

CARACTERIZAÇÃO DA UTILIZAÇÃO DE TÉCNICAS PARA PREVENÇÃO DA EROSÃO ARQUITETURAL NAS EMPRESAS DE TI NO BRASIL

ITABAIANA
2015

UNIVERSIDADE FEDERAL DE SERGIPE
CAMPUS ALBERTO CARVALHO
DEPARTAMENTO DE SISTEMAS DE INFORMAÇÃO

YTHANNA DE OLIVEIRA GOMES

CARACTERIZAÇÃO DA UTILIZAÇÃO DE TÉCNICAS PARA PREVENÇÃO DA EROSÃO ARQUITETURAL NAS EMPRESAS DE TI NO BRASIL

Trabalho de Conclusão de Curso submetido ao Departamento de Sistemas de Informação da Universidade Federal de Sergipe como requisito parcial para a obtenção do título de Bacharel em Sistemas de Informação.

Orientador: *PROF. MSC. MARCOS BARBOSA DÓSEA*

Coorientador: *PROF. DR. METHANIAS COLAÇO JÚNIOR*

ITABAIANA
2015

YTHANNA DE OLIVEIRA GOMES

CARACTERIZAÇÃO DA UTILIZAÇÃO DE TÉCNICAS PARA PREVENÇÃO DA EROSÃO ARQUITETURAL NAS EMPRESAS DE TI NO BRASIL

Trabalho de Conclusão de Curso submetido ao corpo docente do Departamento de Sistemas de Informação da Universidade Federal de Sergipe (DSIITA/UFS) como parte dos requisitos para obtenção do grau de Bacharel em Sistemas de Informação.

Itabaiana, (____ , fevereiro de 2015).

BANCA EXAMINADORA:

_____

Profº Msc.º Marcos Barbosa Dósea
Orientador

_____

Profº Drº. Methanias Colaço Rodrigues Junior
Coorientador

_____

Profº Drº Joseval de Melo Santana
Examinador

_____

Profº Drº. Alcides Xavier Benicasa.
Examinador

AGRADECIMENTOS

Meus agradecimentos,

A Deus, que com sua imensa grandeza me deu força nesta jornada.

Aos meus pais José Tavares e Cordélia pelo amor e por sempre me incentivarem nos caminhos do estudo e do conhecimento.

A minha irmã Yalle por compartilhar minhas alegrias, sonhos, lágrimas e preocupações (principalmente na etapa TCC).

Ao orientado Profº Msc. Marcos Barbosa Dósea pela orientação no caminho da pesquisa, pelas críticas visando meu empenho e pela maestria com que propaga o conhecimento.

Ao meu coorientador Profº Dr. Methanias Colaço Rodrigues Junior pelas importantes considerações que ajudaram a condicionar o trabalho.

A todos meus amigos pelas palavras de incentivo e aos amigos que conquistei durante o curso pelo companheirismo e o ótimo convívio.

Aos desenvolvedores que contribuíram com a pesquisa do meu TCC. Em especial, Eduardo Freitas por disponibilizar o link da pesquisa no site www.qualidadedesoftware.com.br

Por fim, obrigada a todos que de alguma forma me ajudaram a concluir essa etapa!

Epígrafe

"Mantenha seus pensamentos positivos, porque seus pensamentos tornam-se suas palavras. Mantenha suas palavras positivas, porque suas palavras tornam-se suas atitudes. Mantenha suas atitudes positivas, porque suas atitudes tornam-se seus hábitos. Mantenha seus hábitos positivos, porque seus hábitos tornam-se seus valores. Mantenha seus valores positivos, porque seus valores... Tornam-se seu destino."

(Mahatma Gandhi)

RESUMO

A erosão da arquitetura de software acontece com a evolução do sistema e diminui a qualidade do código. Diversas técnicas são propostas na literatura para evitar esse processo, algumas manuais, outras automáticas com utilização de ferramentas. Entretanto não existe um panorama real da utilização dessas técnicas pelas empresas de TI no Brasil, dificultando a definição de investimentos em pesquisa entre as técnicas realmente utilizadas pelo mercado ou melhoria naquelas que são pouco utilizadas. É nesse sentido que este trabalho se propõe a identificar as principais técnicas que avaliam qualidade no código-fonte. Para atender esse objetivo, foi realizado uma pesquisa de campo, por meio da metodologia survey, aplicando um questionário com 12 perguntas, distribuída através da Web, direcionado à população de desenvolvedores das empresas de TI espalhadas no Brasil. O questionário foi acessado por 375 pessoas, sendo que 84,6% responderam o questionário por completo e 14,4% não concluíram as perguntas. Os principais resultados encontrados demonstram que: i) Mais de 85% dos entrevistados conhecem ou usam algum tipo de técnica que analisa a qualidade do código-fonte (manuais ou/e automáticas); ii) Mais de 90% responderam que as principais dificuldades encontradas são a adaptação do uso da ferramenta ao processo de desenvolvimento e a falta de conhecimento sobre as ferramentas; iii) o perfil dos desenvolvedores que possuem maior nível de formação, consideram as técnicas importantes (80%); iv) 70% dos respondentes consideraram que valores limiares das métricas precisam ser ajustados de acordo com o componente da arquitetura analisado e/ou entidade de negócio manipulada; v) 46,11% responderam que o melhor momento de recomendar problemas no código é durante a implementação.

Palavras-chave: Erosão arquitetural, Técnicas, Qualidade de Código-fonte, *Survey*.

ABSTRACT

The erosion of software architecture does the evolution of the system and decreases the quality of the code. Several techniques have been proposed in the literature to avoid this process, some manuals, other automatic with use of tools. However there is a real picture of the use of these techniques by the IT industry in Brazil, making the definition of investment in research between the techniques actually used by the market or improving those that are little used. In this sense, this study aims to identify the main techniques that assess quality in the source code. To meet this goal, we performed a field survey through the survey methodology, applying a questionnaire with 12 questions, distributed through web, directed to the population of developers of IT companies spread in Brazil. The questionnaire was accessed by 375 people, of which 84.6% answered the questionnaire completely and 14.4% did not complete the questions. The main results show that: i) More than 85% of respondents know or use some kind of technique that analyzes the quality of the source code (manual and / or automatic); ii) More than 90% answered that the main difficulties encountered are adapting the use of the tool to the development process and the lack of knowledge about the tools; iii) the profile of developers who have a higher level of training, consider the important techniques (80%); iv) 70% of the respondents considered that the metrics thresholds need to be adjusted according to the analyzed component architecture and / or manipulated business entity; v) 46.11% answered that the best time to recommend problems in code during implementation.

Keywords: architectural erosion, Technical, Source Code Quality, Survey.

LISTA DE FIGURAS

LISTA DE GRÁFICOS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

A arquitetura de Software fornece uma visão holística do sistema a ser construído e os arquitetos de software possuem diversas funções no processo de desenvolvimento de software. Quebrar a complexidade do desenvolvimento em partes mais gerenciáveis; definir funções, interações e dependências para cada componente; e a comunicação entre os componentes e os desenvolvedores são algumas dessas funções.

Apesar de sua importância, Passos (2010) define que muitas empresas não utilizam à arquitetura planejada. Muitas vezes, pela falta de conhecimento dos desenvolvedores, requisitos conflitantes, dificuldades técnicas ou pela pressão de prazo. Esse desvio na arquitetura ocasiona o que chamamos de erosão arquitetural.

Segundo Silva (2012), erosão arquitetural é a deterioração geral da qualidade de um sistema de software durante a sua evolução. Holt (1997) reforça essa ideia através do conceito que a degradação da arquitetura faz com que os benefícios proporcionais por um projeto sejam anulados: Manutenibilidade, Reusabilidade, Escalabilidade, Portabilidade e outros.

Arquiteturas de software que não foram projetadas para acomodar mudanças tendem a se desgastar mais cedo (Silva 2012). Para Bertran (2011), muitos chegam a um nível que é necessário uma reformulação completa do sistema de software. Características como atribuição inadequada de responsabilidade entre os módulos do sistema, alto acoplamento e "cheiros" no código são exemplos de problemas que geram erosão arquitetural.

Dhami (2013) afirma que a erosão é causada muitas vezes por problemas associados ao desenvolvimento. Citando como exemplo: o uso inadequado de ferramentas e métodos; não rastreabilidade de decisões de *design*; alto custo de manuntenção; acúmulo de decisões do *design*; métodos iterativos; falta de documentação; rapidez na correção; pressão de tempo; falta de experiência com desenvolvimento e vaporização das decisões de *design*.

Para evitar o processo de erosão da arquitetura existem várias técnicas disponíveis na literatura, entre elas destacam-se: Ferramentas de Análise Estática do Código *(FONTANA et al., 2011)* e *(SLINGER, 2005);* Linguagens descrição de arquitetura (MEDVIDOVIC; TAYLOR, 2000) e (ALDRICH et al., 2002); Documentação Arquitetural (CLEMENTS et al., 2003) – Arquitetura de Conformidade (PASSOS et al., 2010). As técnicas destacadas podem ser utilizadas para definição e verificação das regras fundamentais da arquitetura de um

software que pretendem atender os requisitos de qualidade interna e externa, aumentando a probabilidade de sucesso em termos de custo e prazo.

1.1 Problemática

Apesar das várias propostas disponíveis, não foi encontrado um panorama sobre a utilização prática dessas técnicas nas empresas de TI no Brasil. Isso pode levar ao investimento de pesquisas em técnicas que são rejeitadas pelos desenvolvedores ou ainda deixar de aperfeiçoar ferramentas e técnicas que são bem aceitas e utilizadas pelas equipes de desenvolvimento.

1.2 Objetivo Geral

Realizar um *survey* com o propósito de levantar informações sobre as técnicas utilizadas para evitar erosões (desvios) arquiteturais em empresas de Tecnologia de Informação do Brasil**.**

1.3 Objetivos Específicos

- Fazer um levantamento bibliográfico sobre Técnicas que evitam erosão arquitetural;
- Fazer um levantamento bibliográfico sobre o processo de aplicação de *surveys*;
- Pesquisar e analisar as principais ferramentas de *survey* que possam ser usadas para aplicação on-line de questionários;
- Elaborar um questionário para coletar informações sobre o uso de técnicas para evitar erosão arquitetural;
- Aplicar o questionário nas empresas do Brasil;
- Analisar usando estatística descritiva os dados obtidos.

1.4 Metodologia do Trabalho

A pesquisa utilizada no projeto foi à exploratória. Segundo Gil (1999) é desenvolvida com o objetivo de proporcionar visão geral, de tipo aproximativo, acerca de determinado

assunto. O produto final desse processo, passa a ser um problema mais esclarecido, passível de investigação.

Em relação ao procedimento para obtenção dos dados foi utilizado o método *survey*. Segundo Babbie (1999) a escolha do modelo de *Survey* a ser aplicado deve levar em conta alguns quesitos, tais como o objetivo da pesquisa e o tempo necessário e disponível para aplicação da pesquisa.

A aplicação do método foi realizada obedecendo às etapas abaixo:

- Etapa 1 – Definição do objetivo e questões da pesquisa;
- Etapa 2 – Identificação do público-alvo;
- Etapa 2 – Elaboração do plano amostral;
- Etapa 3 – Planejamento da amostragem;
- Etapa 4 - Elaboração do pré – teste do questionário;
- Etapa 5 – Aplicação do questionário nas empresas de tecnologia de informação do Brasil;
- Etapa 6 – Análise dos Dados.

A descrição detalhada de cada uma das etapas para realização desta pesquisa é detalhada no Capítulo 3,

1.5 Trabalhos relacionados

Vários trabalhos relacionados investigam aspectos de qualidade através *surveys* em engenharia de software. No entanto, nenhuma dessas pesquisas é focada nas técnicas que evitam erosão na arquitetura e design de software. Não há pesquisas semelhantes encontrados no Brasil, justificando a relevância desta etapa do trabalho. Mas a seguir são relacionados alguns trabalhos que utilizam survey para investigar aspectos que influenciam na qualidade do software.

Uma pesquisa relata em AGNER (2013) um inquérito com 1.740 desenvolvedores e pesquisadores de software sobre as lacunas na compreensão de conhecimentos UML E MDE nas indústrias do Brasil (ou seja, os fatores sociais e organizacionais, diagramas, nível de complexidade e maturidade UML). A maioria dos entrevistados, originados principalmente do

Sudeste, perceberam o valor da modelagem de abordagem UML. Além disso, a pesquisa concluiu que os principais problemas da adoção de UML consistem na falta de competências e instrumentos coerentes. O estudo, no entanto, não foca a área especifica de técnicas para erosão da arquitetura, mas possui um resultado semelhante a nossa pesquisa: a falta de conhecimento técnico para utilizar métodos e ferramentas.

ARCOVERDE et al. (2011) realizou uma pesquisa que afirma que a refatoração ajuda na longevidade do código. Essa pesquisa envolveu cerca de 30 profissionais, sendo divididos em dois grupos. O primeiro questionário, grupos de sujeitos que trabalham mais frequentemente com implementações reutilizáveis e o segundo com aplicações. A pesquisa descobriu que desenvolvedores adiam muito a refatoração dos códigos, pelos seguintes motivos: medo de quebrar o código do cliente ou aplicações derivadas ao realizar as devidas mudanças e pela utilização de ferramentas específicas para refatoramento. A pesquisa objetivou para os resultados finais: identificar melhorias e fraquezas nas ferramentas de refatoração e definir estratégias de refatoração mais eficientes para reutilizar nos sistemas. A pesquisa também chega a uma conclusão semelhante que é o número elevado de profissionais com falta de conhecimento nas ferramentas.

SCHOEFFEL (2010) recolheu experiências na melhoria de processos de software nas empresas do estado de Santa Catarina, com o objetivo de mostrar os resultados visando esses fatores. Essa pesquisa contou com 81 respostas de diferentes empresas da área de software. Os principais resultados desta pesquisa mostram que são utilizadas pouca documentação e métricas no monitoramento dos processos, além de enfatizar a necessidade de aumento de adesão de programas de melhorias aos profissionais da empresa. A comparação dessa análise com a do trabalho, surgiu na etapa da conclusão de utilizarem pouca documentação. A falta de técnicas como forma de evitar erosão, traz também grandes consequências aos sistemas.

Outra pesquisa realizada por ROST (2013) pediu a opinião de 147 participantes de oito países diferentes (Alemanha, Brasil, Finlândia, França, Japão, Suencia, Suiça e Estados Unidos), onde o objetivo principal foi estudar a perspectiva de desenvolvimento em arquitetura de software. O resultado desta pesquisa constatou que a maioria das empresas usam documentação arquitetural ultrapassada, tornando-o na maioria dos casos, menos relevante e útil para os desenvolvedores. A pesquisa também confirma através do survey que os arquitetos precisam de mais apoio em relação às ferramentas para a criação de documentação adequada na arquitetura.

Esse último trabalho vai de encontro com alguns tópicos da pesquisa realizada. Rost (2013) mostrou em sua pesquisa a importância de introduzir ferramentas automáticas nos

trabalhos de documentação, ou seja, concluiu com o mesmo resultado na pesquisa demonstrada nesse trabalho, o mercado necessita do aumento da utilização de técnicas que evitam erosão na arquitetura de software.

1.6 Organização do Trabalho

O restante deste trabalho está estruturado da seguinte forma:

O Capítulo 2 apresenta a **Fundamentação Teórica** sobre principais temas relacionados à pesquisa, a saber: Arquitetura de Software, Erosão e Técnicas que evitam erosão arquitetural e *Survey*.

O Capítulo 3 a **Metodologia**, contém a descrição detalhada do método de pesquisa utilizado do trabalho.

No Capítulo 4, é realizada a **Análise Resultados** obtidos com a pesquisa *Survey*.

O Capítulo 5 encerra com a **Conclusão** e propostas de possíveis trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo aborda os conceitos utilizados para entender os principais temas relacionados à elaboração do *survey* proposto. Para tanto, a Seção 2.1 introduz o conceito de arquitetura de software, na Seção 2.2 são apresentados o conceito de erosão arquitetural e suas características e na Seção 2.3 são detalhadas as principais técnicas para evitar a erosão da arquitetura. A Seção 2.4 apresenta as etapas da pesquisa *Survey* utilizadas como base para realização desta pesquisa.

2.1 Arquitetura de Software

A arquitetura do *software* é uma solução estrutural que atende aos requisitos funcionais, técnicos e de qualidade previamente definidos. Segundo Sommervlille (2011), uma arquitetura quando adequadamente documentada, facilita a compressão da estrutura de um sistema e evita a deterioração do código fonte ou anomalias arquiteturais. Assim, o desenvolvimento do sistema tem melhoria do reuso, manutenibilidade, escalabilidade e outros.

Segundo Garlan (1994), a engenharia de software eficaz, requer alguns itens no projeto da arquitetura: capacidade de reconhecer paradigmas comuns, de modo que as relações de alto nível entre os sistemas possam ser compreendidas; capacidade de construir os novos sistemas com variações de sistemas antigos; compreender a arquitetura, a fim de permitir que o engenheiro possa fazer escolhas de princípios entre as alternativas do projeto; representar a análise e a descrição das propriedades, objetivando se necessário, o uso futuro em sistemas mais complexos.

O processo da arquitetura de software, ainda envolve algumas atividades importantes na criação de uma arquitetura bem elaborada. BASS (2003) classifica os seguintes itens como sendo importantes nessa etapa: criação do modelo de negócio para o sistema, o arquiteto avalia o custo, o público, o tempo de entrega, a interface e as limitações do sistema; a entendimento dos requisitos; a seleção da arquitetura/tecnologia que melhor suporta a implementação; documentar a arquitetura; implementar o sistema baseado na fidelidade das estruturas e dos protocolos da documentação e por fim, mas não menos importante, garantir vigilância constante na fase de manutenção.

Essas atividades envolvem sistemas sob constantes pressões para se adaptar às mudanças de requisitos, tecnologias e abordagens sociais. E ao mesmo tempo, devem continuar a oferecer níveis aceitáveis de desempenho para os usuários. (SILVA, DE; BALASUBRAMANIAM, 2012). As modificações dos sistemas, muitas vezes, são feitas ao longo de um período de tempo, ocasionando danos à integridade estrutural.

Um sistema adequado é aquele que possui uma arquitetura moldada para implementar requisitos e atender o objetivo do sistema. O mesmo pode se tornar inadequado ao longo do tempo devido a erosão. Fenômeno também conhecido como desvio arquitetônico, envelhecimento de software (PARNAS, 1994) ou Erosão Arquitetural (SILVA, DE; BALASUBRAMANIAM, 2012).

2.2 Erosão Arquitetural

Mudanças sociais e ambientais geralmente afetam a arquitetura do software. A arquitetura original deve ser capaz de permitir alterações sem afetar o código-fonte ou apresentar erosões arquiteturais. Fowler (1999) conceitua erosão como problemas relacionados ao uso de más praticas que afetam a evolução do código. O acúmulo de violações de arquitetura e *design* pode tornar o software eventualmente, insustentável e com anomalias denominadas de *code smells*.

2.2.1 Code Smells

O conceito de erosão do software, segundo Fowler (1999), é caracterizado como problemas relacionados ao uso de más praticas que afetam a evolução do código. Conhecido como *Code Smells.*

Code Smells podem influenciar indiretamente na inserção de erros responsáveis por futuras falhas (Fowler, 1999). Em outras palavras, eles são responsáveis pelas dificuldades de manutenção do sistema. Os *Code Smells* são detectados, em geral, por meio de métricas ou técnicas de qualidade que formalizam algumas anomalias no código.

O tratamento de *Code Smells* no código pode ser realizado preventivamente a partir do desenvolvimento do projeto. Para isso é necessária a identificação de suas ocorrências no código, detectando-se assim fragmentos de código que violam a estrutura ou propriedades desejadas.

Fowler (1999) em seu livro, expõe que as erros arquiteturais precisam de refatoração em pequenos passos. O mesmo cita algumas das principais ocorrências conhecidas de erosão arquitetural (*code Smells*) que devem ser tratados. Como: código duplicado, métodos longos, classes grandes, mudanças divergentes, classes preguiçosas e cadeia de mensagens.

Dhami (2013) completa os problemas associados ao desenvolvimento, com: o uso inadequado de ferramentas e métodos; não rastreabilidade de decisões de design; custo de manuntenção; acúmulo de decisões do *design*; métodos iterativos; falta de documentação; rapidez na correção; pressão de tempo; falta de experiência com desenvolvimento e vaporização de decisões de *design*.

O impacto na arquitetura com a erosão é grande e possui um custo associado. Um softwate que não acompanha as mudanças ou apresenta envelhecimento, implica em perda de qualidade ou até mesmo em descontinuidade do mesmo.

Segundo PARNAS (1994) existem dois tipos de envelhecimento de *software*: o primeiro ocorre quando há falhas na adaptação do software para atender os novos requisitos, e o segundo ocorre devido ao resultado provocado pela forma como as mudanças são realizadas. Sistemas envelhecidos podem apresentar perda de desempenho, número crescente de novos erros e perda de usuários devido a concorrência de versões mais recentes.

O diagnostico rápido das erosões arquiteturais em um sistema é importante para que sua remoção seja feita o mais cedo possível no desenvolvimento. Deve-se conhecer quais os mecanismos podem ser aplicados na remoção destes “maus cheiros” do código. A seção seguinte apresenta, ferramentas e técnicas que são utilizados por engenheiros de software, com o intuito de encontrar as principais falhas em seus códigos e evitar o processo de erosão da arquitetura.

2.3 Técnicas Arquiteturais

Para minimizar a erosão arquitetural, existem uma variedade de técnicas e ferramentas propostas na literatura. Estas são divididas em dois grupos: as revisões manuais e as técnicas de análise estática.

A revisão manual é um método antigo para detecção de vulnerabilidade em programas. Para esse tipo de análise o desenvolvedor deve-se possuir noções de arquitetura. A aplicação da técnica não deve ser considerada a solução definitiva ou substituir as outras abordagens, mas sim uma solução complementar.

A segunda é conhecida como análise estática, a mesma consiste na verificação automática do código-fonte. Essa análise geralmente ocorre por meio de ferramentas que analisam o código-fonte em busca de problemas. (CHESS; MCGRAW, 2004). A seção a seguir exemplifica algumas das ferramentas automáticas de revisão de código.

2.3.1 Ferramentas automáticas de revisão

As ferramentas automáticas de revisão do código tem como objetivo realizar verificações no código sem a necessidade da execução do mesmo. Dessa forma é possível antecipar a detecção de vários problemas antes da execução do código. A literatura apresenta uma grande variedade de ferramentas e técnicas para diversas linguagens de programação.

O Checkstyle (BURN, 2014) é uma ferramenta de análise com foco no estilo de codificação e não na lógica ou integridade do código. Essa ferramenta faz checagem de comprimento máximo de linhas em caracteres; convenção de nomes de atributos e métodos; presença obrigatória de cabeçalho; utilização de pacotes e classes importados de outra classe.

Outro exemplo de ferramenta é o PMD (COPELAND et al., 2008) que realiza a análise baseada no código-fonte e também busca por potenciais problemas de código, além de identificar melhorias no sistema. Destaca possíveis trechos com erros; códigos mortos (não utilizáveis) e expressões com alto nível de complexidade.

A Figura 1 exibe a utilização PMD no ambiente Eclipse. Para sua execução deve-se selecionar a classe e clicar com o botão direito -> PMD -> Check Code With PMD. A verificação do código é realizada com base nos módulos configurados na ferramenta. Após a execução é apresentado possíveis erros, alguns códigos mortos e sub-otimizados e expressões de uso complicados.

O PMD possui uma grande variedade de módulos: módulo responsável por verificar se existe concatenação de *Strings* em um laço de repetição; módulo que detecta trechos de código muito aninhados e que dificulta a legibilidade do código; módulo que calcula a complexidade ciclomática de um trecho de código. Além de existir módulos de verificação de métodos excessivamente longos, variáveis curtas e obrigatoriedade do uso de chaves.

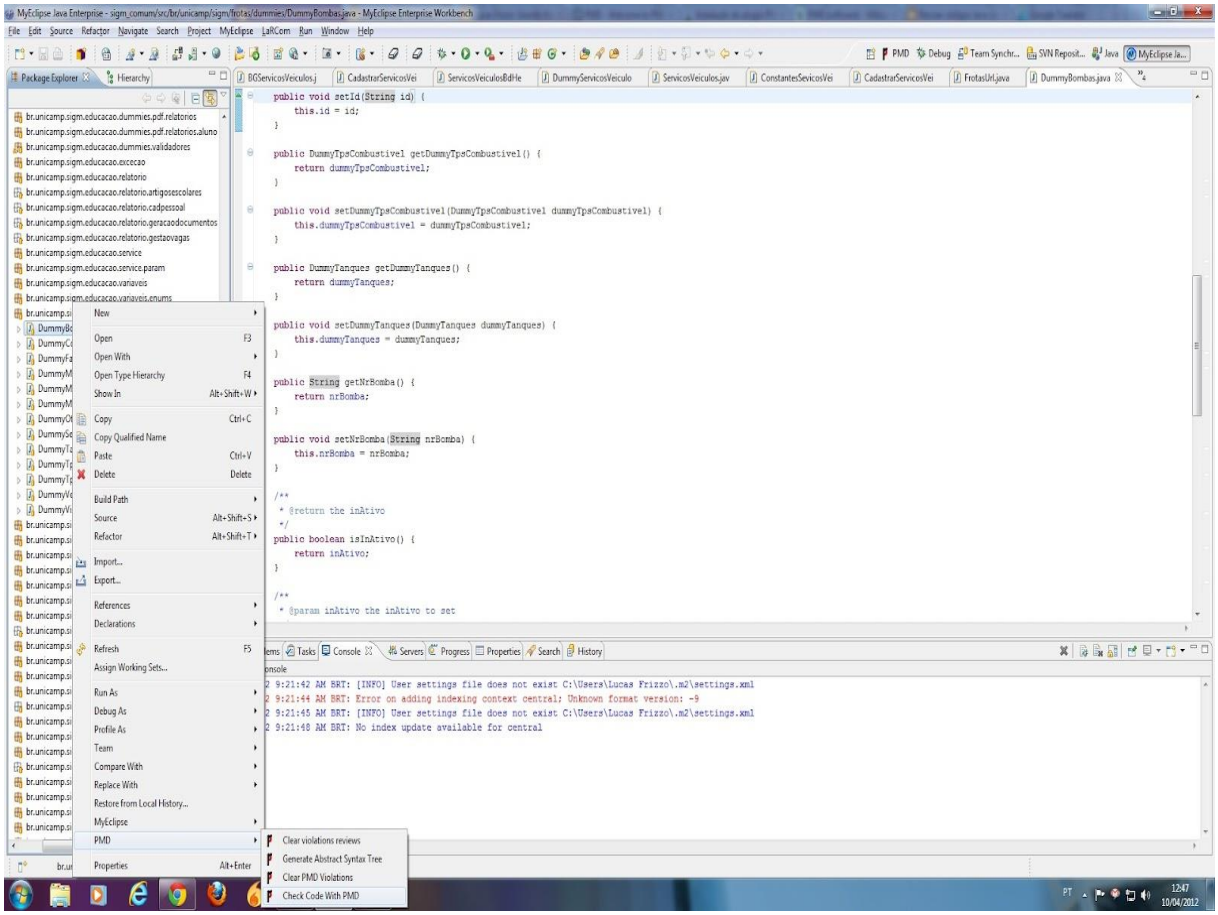

Figura 1 – Utilização da Ferramenta de Análise Estática

2.3.2 Ferramentas que computam métricas do código

Métricas de software devem ser compreendidas como atributos do código-fonte que são medidos, permitindo assim a avaliação do produto. A utilização correta de métricas no ciclo de desenvolvimento de software oportuniza além da qualidade dos produtos, a refatoração.

Basicamente, as métricas mostram dois indicadores básicos muito importantes. O primeiro que a produtividade depende de inúmeros fatores como a dimensão e complexidade dos sistemas a desenvolver, as linguagens de programação, o grau de reutilização ou a experiência e motivação dos participantes no processo de desenvolvimento.

E o segundo indicador, a qualidade dos produtos de software é traduzida através de características como a correção, eficiência, confiabilidade, portabilidade ou facilidade de manutenção. A obtenção desses dados quantitativos relativos a essas características é assim fundamental para introduzir melhorias no processo de desenvolvimento. (ABREU, 1992)

Na literatura também encontram-se várias propostas de ferramentas para cálculo e análise de métricas do código-fonte.

O Kalibro Metrics é uma dessas ferramentas desenvolvida pelo Centro de Competências em Software Livre da USP, deve ser usado em conjunto com outro extrator de métricas. A função da ferramenta é facilitar a interpretação das métricas coletadas, inclusive permitir a criação de limites. Uma vantagem importante da ferramenta é a permissão de exportar os dados para o formato CSV (*comma-separated values*), o que facilita acesso a partir de outras ferramentas.

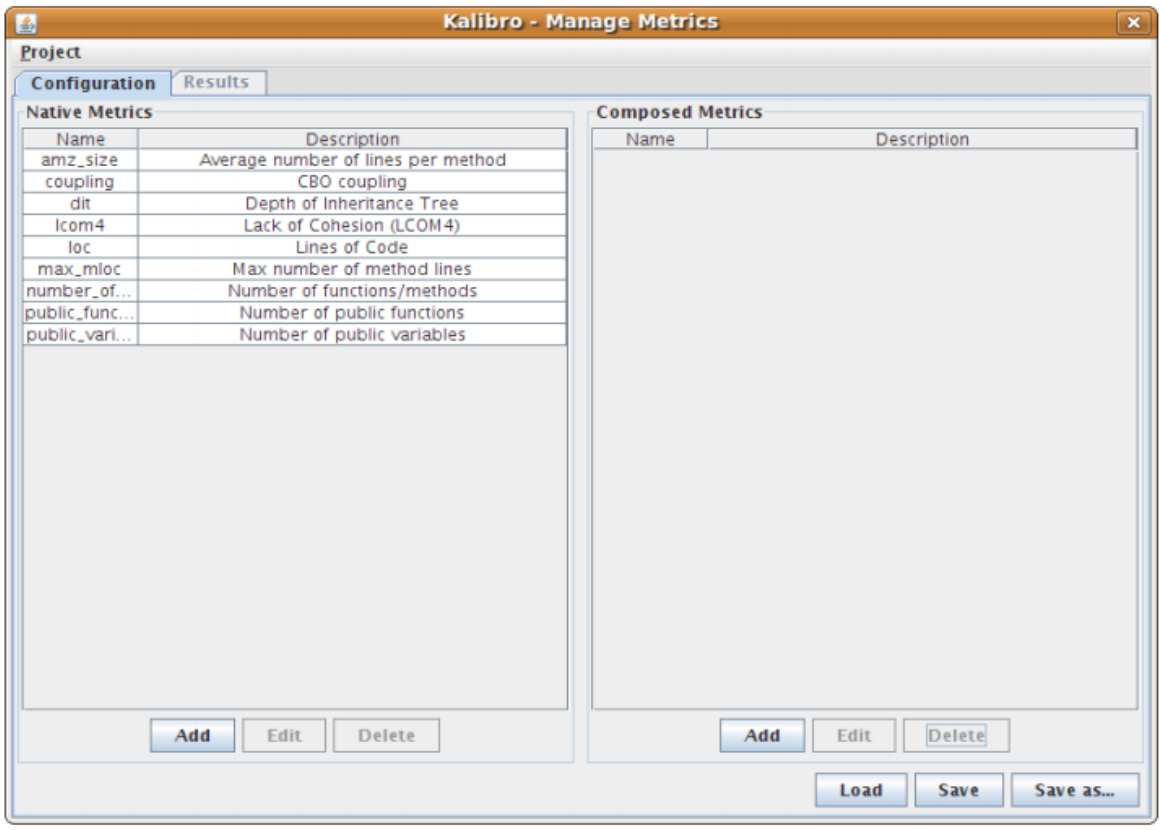

Figura 2 – Tela de configuração do Kalibro Metrics

A Figura 2 mostra a tela de configuração do Kalibro Matrics. Nela é definida a descrição, categoria de contabilização e o peso de cada métrica. Quando um projeto de software é analisado, apenas os resultados das métricas contidas são associadas, com seus valores nos intervalos correspondentes. O usuário também pode criar configurações para atender demandas especificas de cada projeto.

O Sonar é outra ferramenta *open source* popular dedicada ao controle e medição da qualidade do código. Ela possibilita fornecer continuamente métricas capazes de ajudar toda a equipe a identificar falhas e evitar o acúmulo de débito técnico. Tornar visível métricas como

cobertura de testes, complexidade, violações de boas práticas, duplicação de código, nível de comentários, entre outras informações.

Na Figura 3 exibe o *dashboard* do Sonar e o tipo de informações que pode-se visualizar durante todo o tempo. A tela apresenta várias métricas do código, como: número de linhas de código, percentual de comentários, porcentagem de duplicidade no código, listagem de violações a boas práticas de programação (eficiência, portabilidade, usabilidade), cobertura de código em relação aos casos de testes e número de testes realizados com sucesso.

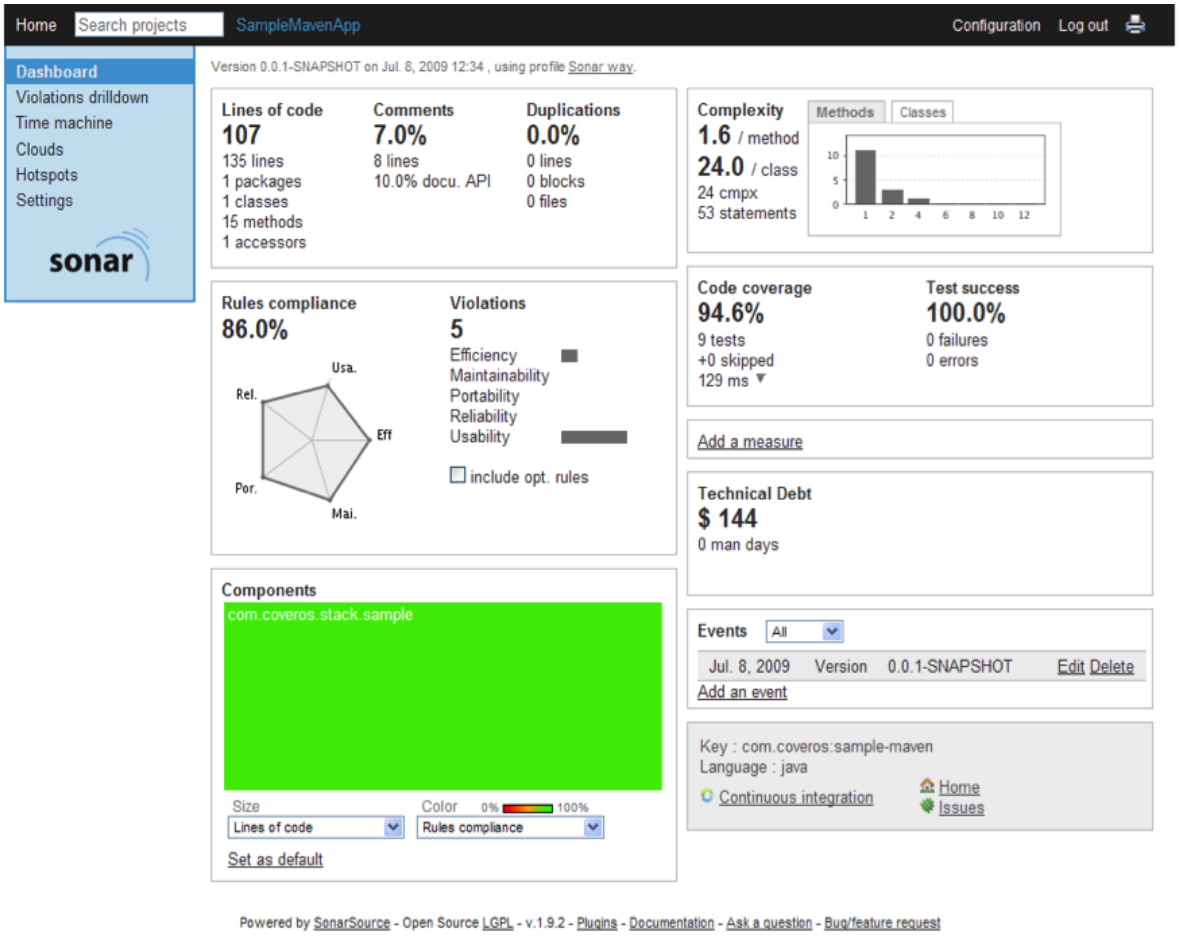

Figura 3 – Sonar

2.4 *Survey*

Nesta seção é detalhado o processo de aplicação de *survey* utilizado como base para realização desse trabalho. As pesquisas são uma adequada estratégia empírica para coletar

dados a partir de uma grande população. Os métodos de pesquisa podem ser quantitativos ou qualitativos, devendo sua escolha estar associada aos objetivos da pesquisa.

O método *Survey* para Mello (2013) "é um método de coleta de informações diretamente de pessoas a respeito de suas ideias, sentimentos, saúde, planos, crenças e de fundo social, educacional e financeiro".

Conforme Kasunic (2005), *Survey* é um método que aborda coleta de dados em diferentes segmentos, tais como: pesquisa documental, experiências de laboratório, pesquisa-ação, estudos de casos, experimentos de Campo, simulação e entrevistas em profundidade.

O método em questão tem como principais finalidades, segundo Babbie (1999):

Descrição - Objetiva descobrir "a distribuição de certos traços e atributos" da população estudada. A preocupação do pesquisador neste caso não é o porquê da distribuição, e sim com o que ela é.

Explicação - Objetiva explicar a distribuição observada. Neste caso, o pesquisador tem a preocupação do por que da distribuição existente.

Exploração - Objetiva funcionar como um mecanismo exploratório, aplicado em uma situação de investigação inicial de algum tema, buscando não deixar que elementos críticos deixem de ser identificados, apresentando novas possibilidades que podem posteriormente ser trabalhadas em um *survey* mais controlado.

2.4.1 Modelos de *Survey*

Em modelos de *Survey*, deve-se levar em contar 2 tipos. Os modelos interseccionais e longitudinais. De acordo com Babbie (1999), o primeiro a coleta de dados é realizada em um único intervalo de tempo. Já o segundo, a coleta é realizada em mais de um intervalo de tempo, possibilitando análise de mudanças e explicações ao longo do tempo.

2.4.2 Processo do *Survey*

A pesquisa *Survey*, por tratar-se de uma pesquisa social empírica, pode se diferir de variadas formas para alcançar o fim ao qual o pesquisador pretende alcançar. A Figura 4 apresenta as principais etapas da metodologia *survey*: identificar o objetivo, identificar e caracterizar público-alvo, elaboração do plano amostral, design e escrita de questionário, teste-piloto do questionário, distribuição do questionário e análise dos resultados.

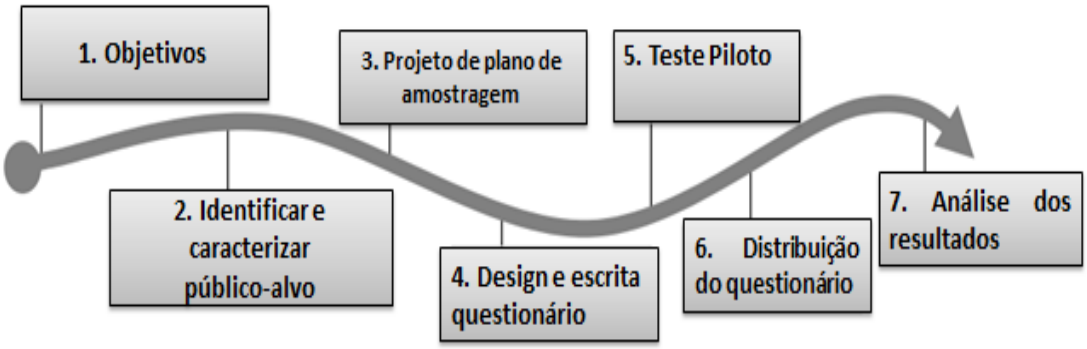

Figura 4 – Etapas de uma pesquisa *Survey* (Figura adaptada de Kasunic 2005)

As subseções a seguir descrevem em detalhes as atividades que devem ser realizadas em cada uma dessas sete etapas.

2.4.2.1 Identificar o Objetivo

A proposta é levantar o escopo e a orientação para o desenvolvimento futuro do questionário. Aqui o interesse está em documentar o que se espera conseguir com a pesquisa. São os objetivos que definem as expectativas razoáveis para o que pode ser realizado em uma única pesquisa (KASUNIC, 2005).

2.4.2.2 Identificar e Caracterizar Público-Alvo

Seguindo o Kasunic (2005) essa etapa começa com a identificação da população que será destinada a responder o questionário. A escolha do público-alvo segue uma base cuja perspectiva vai de encontro com o interesse da pesquisa. Ou seja, para os objetivos da pesquisa identificados anteriormente, qual o melhor público-alvo a responder as perguntas.

A escolha do público-alvo deve ser baseada na perspectiva e interesse que sua pesquisa precisa obter. Ou seja, para os objetivos da pesquisa que são identificados, qual melhor público fornece as informações que necessita?

Depois da identificação do público-alvo da pesquisa, realiza-se uma análise das características do público entrevistado. Essa análise, segue os seguintes entendimentos: A escolha do método de levantamento da pesquisa (papel, entrevista, questionário online); E o desenvolvimento de itens que podem ser interpretados pelos respondentes no questionário (KASUNIC, 2005).

2.4.2.3 Elaboração do Plano Amostral

O objetivo dessa etapa é determinar como os indivíduos serão selecionados para participar da pesquisa, além de planejar o tamanho da amostra necessária. Estas considerações têm consequências significativas quanto à forma e generalização da amostra, assim precisão e confiança serão expressos nos resultados.

Babbie (1999) apresenta dois tipos básicos de amostragem, a probabilística e a não probabilística.

***Amostra probabilística*:** é uma amostra representativa da população da qual foi selecionada se todos os membros da população tiverem oportunidade igual de serem selecionados para a amostra. Freitas (2000) relata que esse tipo "implica utilizar a seleção randômica ou aleatória dos respondentes, eliminando a subjetividade da amostra". A amostra probabilística pode, ainda, ser classificada em amostragem aleatória simples, sistemática, estratificada e por conglomerados em múltiplas etapas.

***Amostra não probabilística**:* é uma amostra onde nem todos os indivíduos têm chances de ser selecionados. A coleta é obtida a partir de algum tipo de critério. Pode ser identificada em seis tipos: por conveniência, mais similares ou mais diferentes, por quotas, bola de neve, casos críticos e casos típicos (Henry apud Bickman e Rog, 1997).

Como foi dito por Kasucic (2005) existem alguns exemplos de amostras não-probabilísticas.

a) Distribuição de questionários para participantes em conferências ou eventos;
b) Especificações de conjuntos de indivíduos em uma área de interesse;
c) Distribuição de questionário utilizando a web permitindo que os indivíduos decidam por si próprio se querem ou não participar da pesquisa.

Ainda, conforme Fink (1995) a amostra também deve possuir preocupação com seu tamanho. Devem-se estabelecer alguns aspectos: o nível de confiança estabelecida e erro permitido; e a proporção em que a característica foco pesquisa se manifesta na população.

2.4.2.4 Planejamento da amostragem (Instrumento)

A escolha da técnica para criação e aplicação do questionário deve-se atentar para o custo, o tempo e a forma que venha garantir uma taxa de resposta aceitável para a pesquisa. E deve-se ser analisada logo na fase inicial dos objetivos e público-alvo. A Figura 5 mostra a

etapa do planejamento. Através dos objetivos e da identificação do público-alvo é planejado a amostragem do projeto.

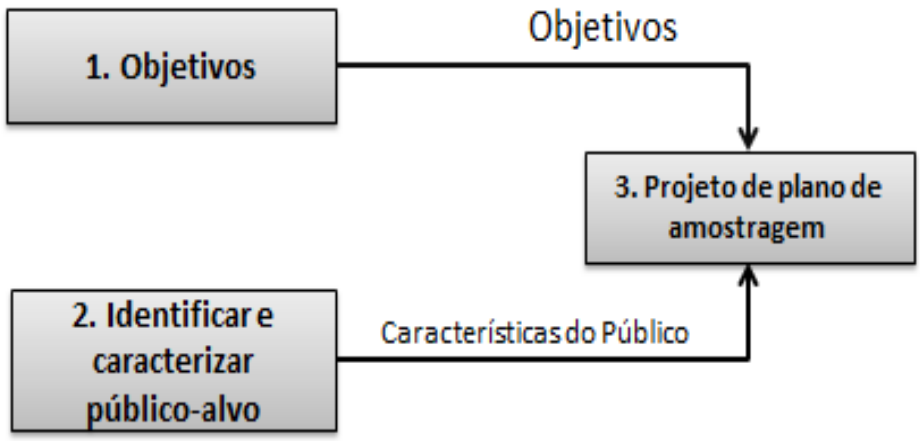

Figura 5 – Etapas do planejamento amostral

Existem algumas etapas na hora de criar os questionários. São elas: Determinar as perguntas; Selecionar o questionário (formato das perguntas); Projetar a sequencia de perguntas e seu *layout*; e Desenvolver documentos anexos.

Freitas (2000) afirma que alguns cuidados devem ser seguidos na elaboração das perguntas. São eles:

- as alternativas para as questões fechadas devem ser exaustivas para cobrirem todas as possíveis respostas;
- somente questões relacionadas ao problema devem ser incluídas;
- deve-se considerar as implicações das perguntas quanto aos procedimentos de tabulação e análise dos dados;
- o respondente não deve se sentir incomodado ou constrangido para responder as questões;
- as questões devem ser redigidas de forma clara e precisa, considerando o nível de informação dos respondentes;
- as questões devem possibilitar uma única interpretação e conter uma única ideia;
- o número de perguntas deve ser limitado;
- as perguntas não devem induzir as respostas;
- a apresentação gráfica do questionário deve ser observada, procurando-se facilitar o preenchimento;

- deve haver um cabeçalho que informe o objetivo da pesquisa e deve haver instruções sobre como preencher corretamente o questionário.

Visto os tipos de categorias, o processo de desenvolvimento da pesquisa segue etapa da elaboração externa do questionário. A extensão do questionário (quantidade de questões); organização e sequenciamento de perguntas (iniciando com as mais simples e seguindo com as mais complexas); instruções e introduções do questionário (explicar de forma resumida o objetivo da pesquisa) e opções de layout de página (escrita ou web). São exemplos que devem ser observados na construção do instrumento.

A opção de layout *web* foi à escolhida para esta pesquisa. Kasunic (2005) descreve a taxa de resposta mais rápida, a facilidade de envio de lembretes para os participantes, a facilidade de processamento de dados e o uso de caixas suspensas, como sendo as principais vantagens do uso do questionário web.

2.4.2.5 Teste-Piloto do Questionário

O Teste-Piloto segundo Babbie (1999) tem o objetivo de melhorar o instrumento da pesquisa. Devem-se avaliar alguns critérios na hora de aplicar o pré-teste, são eles:

- Existe confusão de entendimento nas instruções?
- As perguntas são compreensíveis?
- A ordem das perguntas parece logica?
- Há termos não compreendidos ou com múltiplas interpretações?
- O layout da página tem boa legibilidade?
- Quanto tempo o entrevistado levou para responder o questionário?

Kasunic (2005) cita algumas determinações que devem ser analisadas quando o questionário é baseado na web. A avaliação deve ser seguindo as seguintes recomendações: observar se a entrada de dados não possui nenhum problema; avaliar se os entrevistados conseguiram “navegar” na página sem dificuldade; observar se as mensagens de erro apresentadas no questionário são simpáticas e prestativas; analisar se as entradas de dados ficaram corretamente armazenadas no banco de dados e se foram salvas corretamente.

Segundo Couper (2000), o desenho do questionário quanto implementado em meio eletrônico é substancialmente mais importante do que na implementação em papel. Desta forma, duas considerações são importantes: (1) existem mais ferramentas disponíveis para o pesquisador a um baixo custo, que possibilitam a inclusão de cores, sons, imagens e

animação; (2) a aparência do questionário visualizada pelo respondente pode variar de acordo com a configuração do software, sistema operacional e variações no hardware do equipamento do respondente

Através da resolução dos quesitos abordados nessa seção, o questionário passará por uma melhoria e assim dará uma vantagem inicial na análise final.

2.4.2.6 Distribuição do questionário

A distribuição do questionário acontece depois das devidas correções (se necessário) do teste-piloto. De acordo com Punter, Ciolkowski, Freimut, & John (2003) existem varias maneiras de realizar pesquisas. E são classificados de acordo com o meio que é aplicado durante a coleta de dados. O meio utilizado para o levantamento foi o levantamento através do meio web.

A distribuição da pesquisa por meio web é considerada semelhante à pesquisa realizada com autopreenchimento. Esse tipo de pesquisa geralmente utiliza meios distintos de envio (Silva, Santos e Siqueira, 1997):

a) Envio do questionário por e-mail: O questionário é enviado diretamente para o participante da pesquisa. O mesmo pode ser enviado como um arquivo anexo ou no próprio corpo da mensagem com o link da pesquisa;
b) Disponibilização do questionário em páginas de internet e redes sociais: o entrevistado na área é informado da página do questionário na Internet.

Depois da distribuição do questionário por meio de link ou anexo, deve-se acompanhar a taxa de respostas. Essa taxa ajudará a saber se sua pesquisa está sendo bem aceita ou se precisará do envio de lembretes (Kasunic, 2005). Na Figura 6 é apresentada um gráfico de taxa de resposta que possui lembretes enviados aos destinatários por email. Nota-se que cada envio de lembretes aumenta imediatamente o número de questionários concluídos.

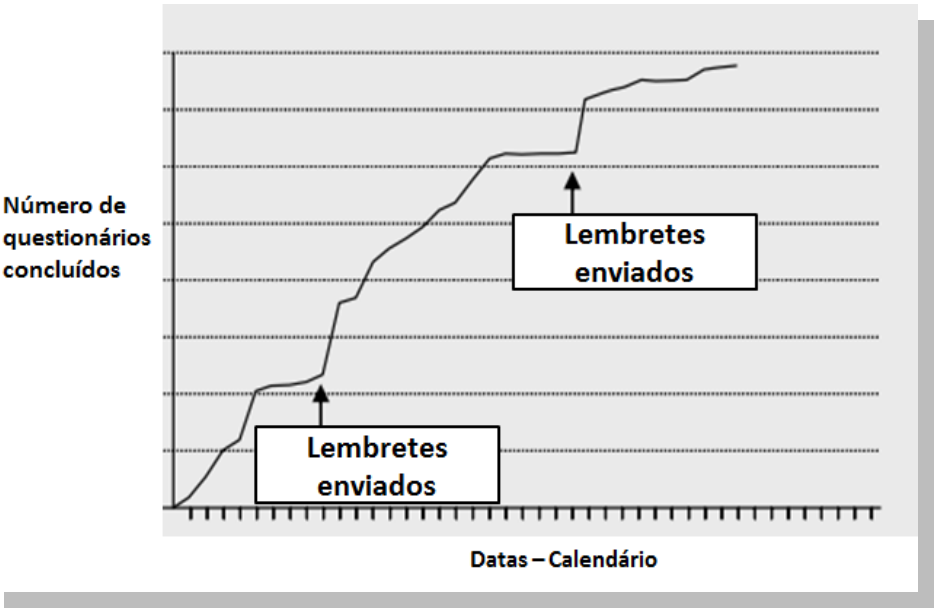

Figura 6 – Taxa de resposta

2.4.2.7 Análise dos Dados

A análise dos dados refere-se ao processamento dos dados coletados através do instrumento de pesquisa.

Assim, Kasunic (2005) aponta dois tipos de variáveis na obtenção dos resultados: o quantitativo e a qualitativo. A primeira objetivando número em determinada escala e a segunda com resultados com atributos e qualidades. Além das variáveis, existem tipos de atribuições que são classificadas em nominal, ordinal, intervalar e de razão.

O levantamento em geral serve para pesquisas descritivas que pretendem dizer através da população selecionada, quantas pessoas têm determinados atributos, ou até mesmo explorar aspectos de uma situação, procurar explicações, entre outros.

Os dados são analisados de forma conjunta e descritiva, sem precisar das identificações dos profissionais nas respostas. Segundo Marconi e Lakatos (1999) a pesquisa descritiva: “Delineia o que é abordando “… quatro aspectos: descrição, registro, análise e interpretação de fenômenos atuais, objetivando o seu funcionamento no presente”. Todos esses passos citados anteriormente foram analisados na pesquisa online.

A análise a ser realizada depende do tipo de variável utilizada (FREITAS et al., 2000):

a) *Variável nominal*: tipo de variável mais simples, os elementos do conjunto são agrupados em classes ou categorias;

b) *Variável ordinal:* resulta na operação de ordenar postos;

c) *Variável intervalar*: possui características das duas variáveis citados acima, além de apresentar distâncias iguais entre os intervalos que se estabelecem sobre medida.

Um dos principais benefícios do uso de questionários baseados na Web é a facilidade de coleta de dados e organização de dados: é tudo automatizado. Obtém-se um resultado em tempo real. Os dados são apresentados em forma de gráficos, para assim desenvolver uma análise das conclusões preliminares.

3 PESQUISA: APLICAÇÃO DO SURVEY

Como descrito na Seção 1.2, o método de pesquisa empregado para realização do trabalho foi o *Survey*. Assim, para que fosse possível obter os resultados almejados, foram elaborados e aplicados questionários para o público-alvo de desenvolvedores de software. As subseções a seguir apresentam o percurso metodológico da pesquisa, baseado nas etapas detalhadas na Seção 2.4.

3.1 Identificar o Objetivo

Segundo Kasunic (2005), na primeira etapa da pesquisa *survey* deve-se "identificar os objetivos" e determinar suas questões da pesquisa. O objetivo definindo foi: investigar o uso de técnicas para prevenção da erosão (desvios) arquitetural no design do software em empresas de tecnologia de informação do Brasil. Foram definidas cinco questões de pesquisa:

Q1) Alguma técnica manual ou automática é utilizada para análise do código?

Q2) Quais os níveis de importância dado para utilização das técnicas?

Q3) Quais as dificuldades para utilização das técnicas automáticas?

Q4) O contexto da classe dentro da arquitetura deve ser considerado pelas técnicas de análise?

Q5) Qual o melhor momento para aplicação das técnicas automáticas?

O *survey* tem caráter exploratório, isto é, por meio deste estudo, pretende-se obter informações que permitam realizar uma caracterização inicial efetiva das técnicas utilizadas pelos desenvolvedores na empresas de TI do Brasil. Além disso, sua aplicação foi realizada de forma não-supervisionada, ou seja, os participantes responderam ao *survey* seguindo as instruções descritas, sem supervisão por parte de terceiros.

3.2 Identificar e Caracterizar o Público-Alvo

Na segunda etapa da pesquisa *survey*, o foco principal é o público-alvo. Analisando as questões da pesquisa e seu objetivo, contatou-se que o grupo alvo dessa pesquisa são os profissionais que trabalham na área de desenvolvimento de software no Brasil, enfatizando principalmente o envio aos analistas, programadores de software e engenheiros. Os

profissionais foram convidados a participar da pesquisa de forma online, tendo a liberdade de responder ou não o questionário.

3.3 Projeto do Plano Amostral

Conforme apresentado na Seção 2.4.2.3, o terceiro passo na elaboração do instrumento de um *survey* é o Plano amostral. A complexidade desta etapa é emoldurar uma amostra que seja viável e que tenha uma taxa de retorno de respostas.

O público-alvo da pesquisa foram profissionais de desenvolvimento de software no Brasil. A utilização de uma amostra probabilística tornou-se inviável pela falta de informações sobre o número de profissionais que atuam na área de desenvolvimento de software no Brasil.

Por esse motivo, optou-se por uma amostra não-probabilística, ou seja, as descobertas serão consideradas unicamente por aqueles que participaram do *survey.*

3.4 Design e Escrita do questionário

O questionário foi planejado seguindo as recomendações de Kasunic (2005) e Babbie (1999). As questões foram agrupadas por pertinência de assunto. Cada questão seguindo uma ordem de dificuldade, iniciando-se com as perguntas mais fáceis e rápidas de responder. Assim o destinatário responderia com mais facilidade e não abandonaria o questionário logo no início.

A quantidade de perguntas deve ser um ponto importante de avaliação do pesquisador. Buscou-se construir um questionário com um número mínimo de perguntas para atender aos objetivos. Define-se apenas doze questões para que proporcionasse um maior retorno de respostas.

Para que o instrumento esteja adequado às suas medições, é necessário definir as escalas utilizadas nas perguntas. De acordo com a seção 2.4.2.4, os tipos de perguntas escolhidas foram às fechadas (múltipla escolha) e abertas (caixa de texto). O questionário foi divido em 3 partes: a primeira foi relacionada as perguntas que correspondem as técnicas. Essas foram divididas em 5 questões de pesquisa:

Para a primeira questão de pesquisa (RQ1) foi elaborada uma pergunta.

RQ1: Alguma técnica manual ou automática é utilizada para análise do código? **01**) Quais técnicas são utilizadas para análise da qualidade do código fonte desenvolvido pela empresa? Mais de uma opção pode ser selecionada.

( ) Revisões manuais do código
( ) Ferramentas automáticas de revisão (Ex: Checkstyle, PMD, FindBugs, FxCop)
( ) Ferramentas que computam métricas do código
( ) Ferramentas desenvolvidas pela própria empresa para análise do código
( ) Outra _____
( ) Nenhuma técnica é utilizada

Para a segunda questão de pesquisa (RQ2) foram elaboradas três perguntas.

RQ2: Quais o nível de importância para utilização das técnicas?
02) Qual o nível de importância atribuído por você para manutenção da qualidade do código da aplicação?
a) Muito importante
b) Importante
c) Pouca importância
d) Nenhuma importância

03) Qual o nível de importância atribuído pela empresa para manutenção da qualidade do código da aplicação?
a) Muito importante
b) Importante
c) Pouca importância
d) Nenhuma importância

04) Qual o periodicidade de revisão da qualidade do código desenvolvido?
a) Pelo menos 1 vez na semana
b) Pelo menos 1 vez ao mês
c) Não há período definido para revisões
d) Nunca são revisados

Para a terceira pergunta de pesquisa (RQ3) foi elaborada uma pergunta:

RQ3: **Quais as dificuldades para utilização das técnicas automáticas?**
05) Quais as principais dificuldades na utilização de ferramentas automáticas para análise da qualidade do código? Mais de uma opção pode ser selecionada.
( ) Avaliar e interpretar os resultados emitidos pelas ferramentas
( ) Definir parâmetros de qualidade (Ex: número máximo de linhas de código por método)
( ) Definir regras da arquitetura da empresa (Ex: regras de comunicação entre camadas)
( ) Dificuldade para adaptar o uso da ferramenta ao processo de desenvolvimento.
( ) Falta de conhecimento sobre essas ferramentas
( ) Outra _____

Para a quarta questão de pesquisa (RQ4) também foi definida apenas uma pergunta:

RQ4: O contexto da classe dentro da arquitetura deve ser considerado pelas técnicas de análise?
Questão 06) As ferramentas permitem ajustar valores dos parâmetros utilizados para análise da qualidade do código, por exemplo, o número máximo de linhas em um método ou nível máximo de complexidade de um método. Considerando um sistema em três camadas (GUI, Negócio e Persistência) qual a sua opinião em relação aos valores desses parâmetros para cada camada?
a) Devem ser iguais, afinal os métodos de cada camada possuem códigos com as mesmas características.
b) Devem ser diferentes e definidos de acordo com a camada analisada, já que métodos de uma camada podem possuir diferentes características em relação as demais camadas.
c) Devem ser diferentes e definidos de acordo com a entidade de negócio manipulada, já que métodos que manipulam diferentes entidades podem possuir diferentes características.
d) Letras (b) e (c) devem ser consideradas.
e) Não tenho opinião formada sobre o assunto.

Finalmente para quinta questão de pesquisa (RQ5) foi definida apenas uma pergunta:

RQ5: **Qual o melhor momento para aplicação das técnicas automáticas?**
Questão 07) Na sua opinião, qual seria o melhor momento e formato para recomendar ao desenvolvedor possíveis problemas de qualidade no código alterado?
a) Durante a implementação do código gerando um listagem de recomendações que não interfira no processo de digitação do código.
b) Após a gravação do código no sistema de controle de versão gerando uma lista de recomendações.
c) No momento definido pelo desenvolvedor onde manualmente ele solicita a análise de código e uma lista de recomendações é disponibilizada.
d) Outra_____

A segunda parte foi trabalhada as características do respondente. Nessa etapa, foram elaboradas 5 perguntas. As mesmas identificaram o nível de formação, o tempo de experiência, a posição na empresa e a região que trabalha. A última parte foi finalizada com um espaço aberto para inserir o e-mail. A questão era opcional, caso o respondente tivesse interesse da obtenção dos resultados finais da pesquisa.

Caracterização do Respondente

Questão 08) Qual o seu nível de formação?

a) Técnico
b) Graduação
c) Especialização
d) Mestrado
e) Doutorado

Questão 09) Qual a sua posição atual na empresa?

a) Programador
b) Analista de Sistemas
c) Arquiteto de Software
d) Analista de Qualidade
e) Outra_____

Questão 10) Quanto tempo de experiência possui na área de desenvolvimento?

a) < 02 anos
b) 02 a 05 anos
c) 05 a 10 anos
d) > 10 anos

Questão 11) Quantos sistemas já desenvolveu ou executou atividades de manutenção?

a) < 03 sistemas
b) 03 a 05 sistemas
c) 06 a 10 sistemas
d) > 10 sistemas

Questão 12) Região do Brasil onde trabalha atualmente?
a) Norte
b) Nordeste
c) Centro-Oeste
c) Sudeste
d) Sul

Com o questionário preparado para inicialização do envio, buscou-se serviços de *survey* na internet, que permitissem o envio de um número considerável de questionário, e

posteriormente permitisse a exportação dos dados de respostas para que pudessem ser analisados.

3.5 Teste-Piloto

A fim de realizar uma avaliação prévia do *survey,* de forma a permitir a identificação de refinamentos necessários e adequação do instrumento de pesquisa, foi realizado um piloto. O mesmo foi criado na ferramenta SurveyMonkey (SurveyMonkey Inc., 2015) e aplicado no mês de dezembro de 2014, com 6 desenvolvedores de software. Estes possuíam conhecimento no assunto, por conseguinte, contribuíram de forma significativa na validação do questionário.

Os "pilotos" realizados com os profissionais da área, tiveram como resultados avaliativos uma boa adequabilidade das questões propostas ao foco central do estudo. O teste também contou com a avaliação do tempo de resposta, tornando-se ativo a média de 5 minutos estabelecidos no planejamento de amostragem.

Com base nos comentários recebidos, o questionário foi ajustado em relação a ordem de questões e termos utilizados. O questionário não sofreu grandes alterações, sendo adequado a distribuição aos profissionais e empresas da área de software no Brasil.

3.6 Distribuição do Questionário

A presente pesquisa optou pelo envio através da web, considerando principalmente o fato de ser realizada com desenvolvedores de software. Com base nessa escolha, foi analisada algumas ferramentas de *survey* na internet, que permitissem o envio de um número considerável de questionários, e posteriormente permitisse a exportação dos dados de resposta para que pudessem ser analisados.

3.6.1 Ferramentas Online de *Survey*

Inicialmente foi realizada uma busca das principais ferramentas online com serviço de survey. Agrupou-se as principais vantagens e desvantagens comuns entre algumas ferramenta. São elas: SurveyMonkey, Zoomerang e QuestionPro.

As três ferramentas selecionadas para avaliação foram analisadas na versão gratuita disponibilizadas em seus sites. Os critérios mais importantes avaliados para a escolha da

ferramenta foram: tipos de questões, armazenamento e manutenção dos dados e análise dos dados.

A Tabela 1 mostra as principais características de três ferramentas *survey* online selecionadas. Em relação a quantidade máxima de questionários, a surveyMonkey possui um leque maior de modelos disponíveis. Já no quesito quantidade máxima de perguntas a zoomerang tem valor máximo de 20. E no fator quantidade de aplicações a QuestionForm lidera das outras, pelo fato de ser ilimitada.

Nos itens que são de grande relevância para a pesquisa a SurveyMonkey saiu a frente das outras ferramentas analisadas. Ela permite a coleta de dados por meio de links enviados por email, própria ferramenta e redes sociais. Além disso na análise dos dados a surveyMonkey mais uma vez apresentou melhores características. Todas só apresentam gráficos estatísticos no formato pago, mas a surveyMonkey permite no formato gratuito a edição e inclusão de novas respostas.

Análise Comparativa: Ferramentas			
	FERRAMENTAS		
CARACTERÍSTICAS	Survey Monkey	Zoomerang	Question Form
Tipos de questões	15	14	10
Quantidade máxima de questionários	ilimitado	ilimitado	ilimitado
Quantidade máxima de perguntas por questionário	10	20	5
Quantidade de aplicações permitidas de cada questionário	100	100	ilimitado
Meio de Coleta de Dados			
Link enviado por email pelo usuário	Sim	Sim	Sim
Link enviado por email pela ferramenta	Sim	Sim	Não
Compartilhamento via Redes Sociais	Sim	Sim	Não
Análise dos Dados			
Apresenta gráfico para as respostas	Não	Não	Não
Permite edição das respostas	Sim	Não	Não
Permite inclusão de novas respostas	Sim	Não	Não

Tabela 1 – Análise comparativa das ferramentas

Diante das considerações acima, optou-se pela ferramenta SurveyMonkey. A mesma atendeu todas as necessidades e características da pesquisa. E para um melhor proveito de todas as funcionalidades, decidiu-se adquirir uma licença do plano PLUS (24 dólares/mês), cuja aquisição incrementava número de respondentes que poderia acessar o questionário. Além disso outras características da ferramentas foram importantes para escolha:

- Layout amigável e fácil de usar, tanto para quem cria o questionário quanto para os entrevistados;

- Exportação de resultados com formato simples e mais complexos entre gráficos.

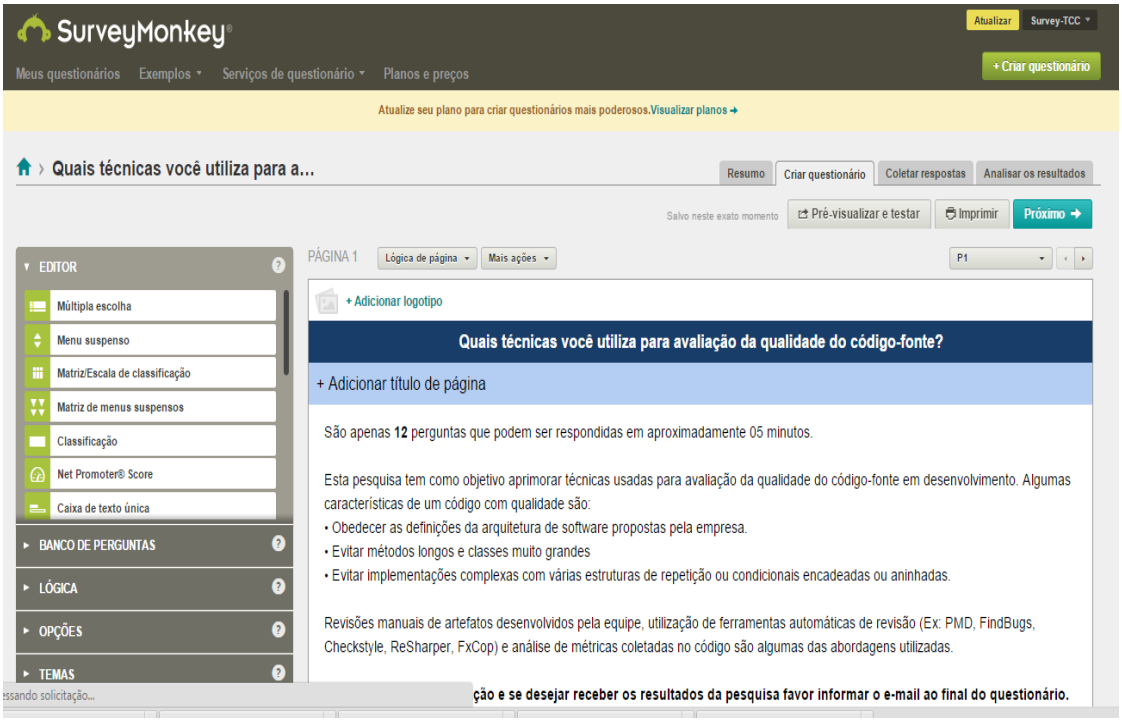

Figura 7 – Ferramenta SurveyMonkey

A Figura 7 mostra o layout de configuração da ferramenta SurveyMonkey. Nela é encontrada o editor, o banco de perguntas, a lógica para as respostas, opções de matriz e classificação e várias outras funcionalidades que ajudam a construção do instrumento (o questionário).

Posteriormente, com a ferramenta escolhida, o questionário oficial foi distribuído por meio de um link. Foram utilizados no estudo, diferentes meios de divulgação. Entre eles foi realizada a divulgação através do site www.qualidadesoftware.com. Este site tem um grande público na área de qualidade e o mesmo foi de grande importância na divulgação da pesquisa. Além do site, a distribuição aconteceu por meios de grupos (Google, Facebook e Linkedln) e de forma direta por meios de mensagens *inbox* nas redes sociais.

Cabe aqui um ponto de destaque, que foi taxa de retorno. Mais de 85% das pessoas que receberam o link responderam a pesquisa, principalmente nas primeiras semanas. Como mostra a figura 8.

A pesquisa foi encerrada após a quantidade de questionários ter alcançado um número superior a 300 (trezentos). A ferramenta recebeu 375 acessos ao questionário,

respondidos de diferentes profissionais que possuíam conhecimento básico ou avançado de técnicas para avaliar a qualidade do software.

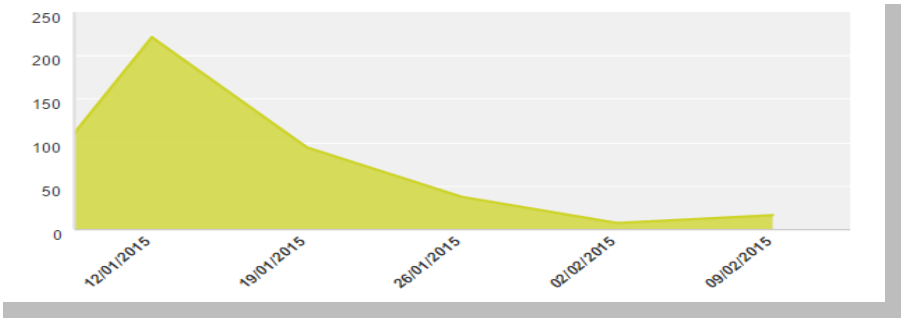

Gráfico 1 – Taxa de Retorno

O Gráfico 1 mostra que a maior parte dos respondentes acessou o questionário na primeira semana. Essa alta taxa deve-se pelo fato de ter iniciado a pesquisa através dos emails e pela divulgação no site de qualidade de software.

3.7 Análise dos dados

A partir dos dados coletados nos questionários foram realizadas análises sobre as respostas colhidas para cada pergunta. Por meio de tal processo, foi possível selecionar desenvolvedores e pesquisadores de software: através de e-mails, redes sociais e fóruns em grupos. A pesquisa foi realizada no período de dezembro de 2014 até 2 fevereiro de 2015. Assim, após um período de levantamento, um total de 375 acessaram a pesquisa, acessando a pesquisa pelo link da ferramenta surveyMonkey.

Apenas um subconjunto dos participantes que iniciaram o levantamento não concluiu o inquérito. Dos 375 que acessaram a pesquisa, apenas 14,4% não concluíram todas as perguntas. Ou seja, a amostra total obteve 85,6% de taxa de conclusão. Consideramos que os fatores que contribuíram para a baixa porcentagem de desistência foram: a escolha de perguntas menos complexas inseridas no inicio do questionário, a informação do tempo para realização do questionário, utilização de uma linguagem próxima do público-alvo, o pequeno número de perguntas e o assunto abordado na pesquisa.

Os resultados da aplicação do questionário e análises realizadas para responder as cinco questões de pesquisa são apresentadas no Capítulo 04 deste trabalho.

4 RESULTADOS

Neste capítulo são apresentados os resultados que têm por intuito analisar os respondentes da pesquisa. A Seção 5.1 faz uma análise do perfil dos respondentes do questionário. As cinco subseções seguintes apresentam análises sobre as cinco questões de pesquisa, apresentadas na Seção 4.1, e consideradas nas análises realizadas.

4.1 Análise dos respondentes

Nesta seção são apresentados resultados sobre o perfil dos respondentes do questionário. Foram cinco perguntas para caracterização do perfil cujos dados são exibidos nos gráficos abaixo. As perguntas tinham como objetivo coletar dados relativos ao nível de formação dos entrevistados, a posição atual na empresa que trabalha, ao tempo de experiência na área de desenvolvimento, a quantidade de números de sistemas que desenvolveram manutenção e por fim, a região que trabalha.

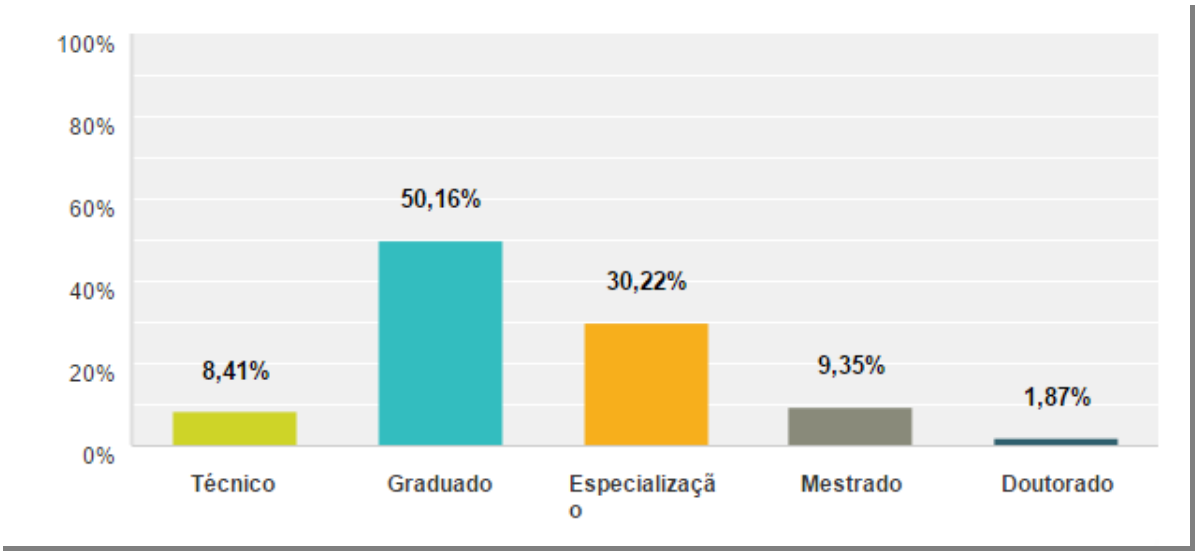

Gráfico 2 – Nível de formação dos entrevistados

Foi identificado no gráfico 2 o nível de formação do entrevistados, a grande maioria possui a formação de graduado (50,16%). Também houve um número considerável de especialistas (30,22%), enquanto (9,35%) mestrado e (1,87%) doutorado. Apenas 8,41% dos respondentes possuíam apenas nível técnico, observa-se dessa forma bom nível de formação dos respondentes, onde mais de 91% possui formação pelo menos de graduação. Ou seja, 91, 6% dos respondentes possuem n

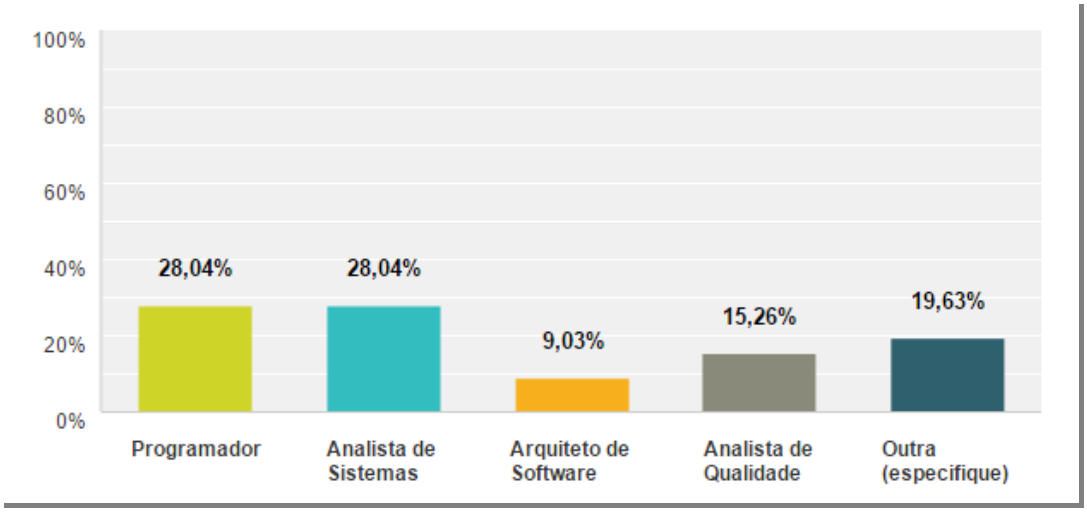

Gráfico 3 – Posição atual do entrevistado na empresa

Em relação a posição atual na empresa, apresentada no Gráfico 3, os resultados dessa pergunta de múltipla escolha obteve os seguintes dados: Tanto programador como analista de sistemas obtiveram (28,04%) de respostas, (15,25%) responderam trabalham como analista de qualidade, (9,03%) como arquiteto de software e (19,63) outros. Esse último item obteve as seguintes especificações de respostas: Engenheiros de Software, Líder Técnico, Gerente de projetos, diretor de desenvolvimento, diretor e coordenador de qualidade, todas as áreas ligadas a área de desenvolvimento de software, público-alvo da pesquisa.

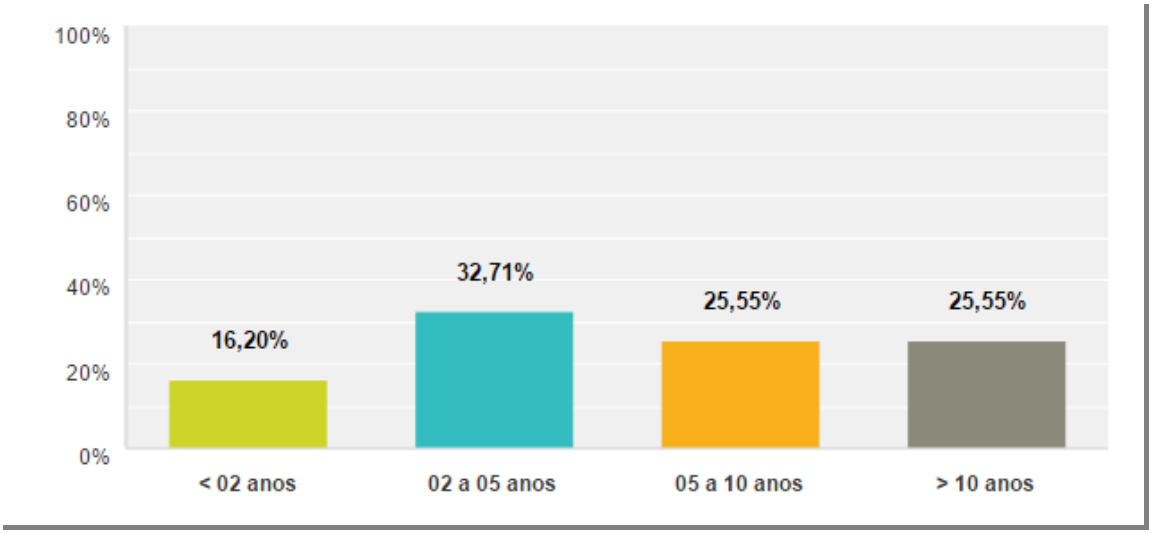

Gráfico 4 - Tempo de experiência na área de desenvolvimento.

O gráfico 4 mostra os resultado dos níveis de experiência na área de desenvolvimento. Vê-se que (32,71%) dos desenvolvedores possuem entre 2 a 5 anos de experiência em desenvolvimento, enquanto (25,55%) possuem de 5 a 10 anos, mais de 10 anos (25,55%) e apenas (16,20%) tem como resposta menos de 2 anos de experiência na área.

Foi realizada uma correlação entre o nível de experiência e posição atual na empresa, apresentada no Gráfico 5. Percebe-se que os gerentes de projetos e diretores, que responderam a opção outros, estavam entre os participantes com “mais de 10 anos” de experiência de trabalho (30,49%). Os profissionais com menor nível de experiência possuem normalmente a posição de programador (38,46%). E acima de cinco anos de experiência as posições na empresa são de analistas de sistemas, arquiteto de software e outras.

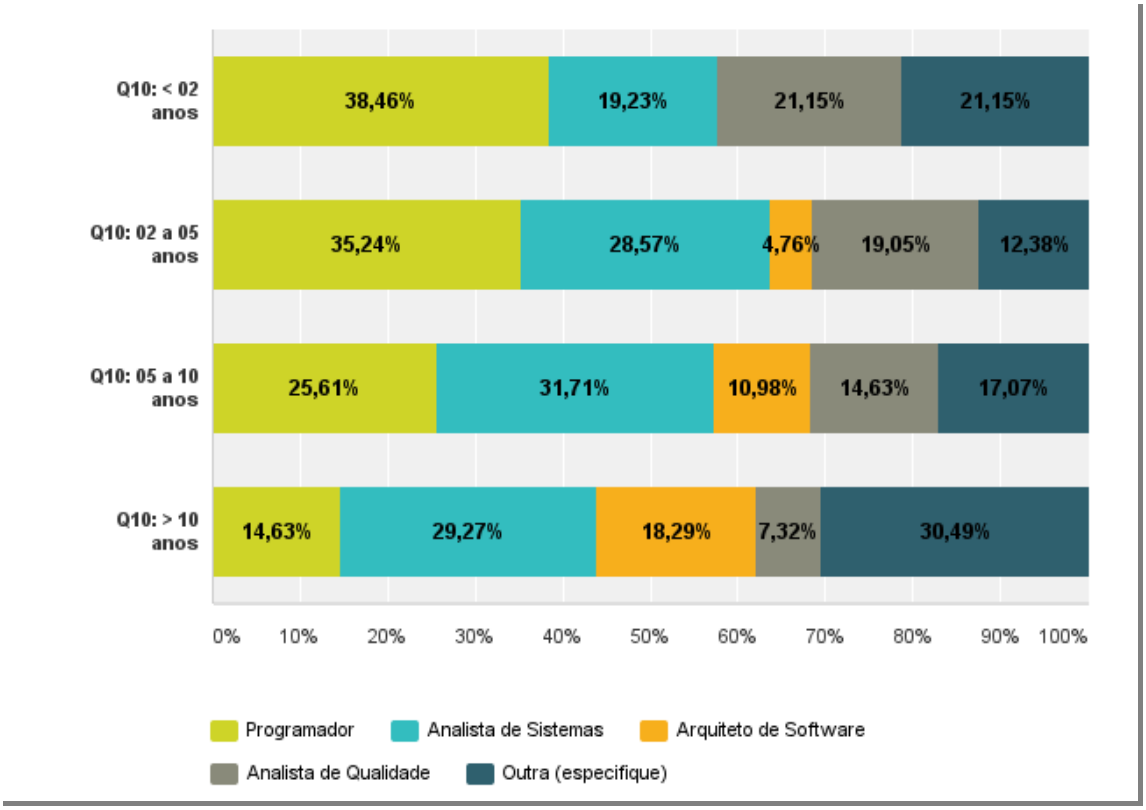

Gráfico 5 – Comparativo da posição atual da empresa X tempo de experiência na área

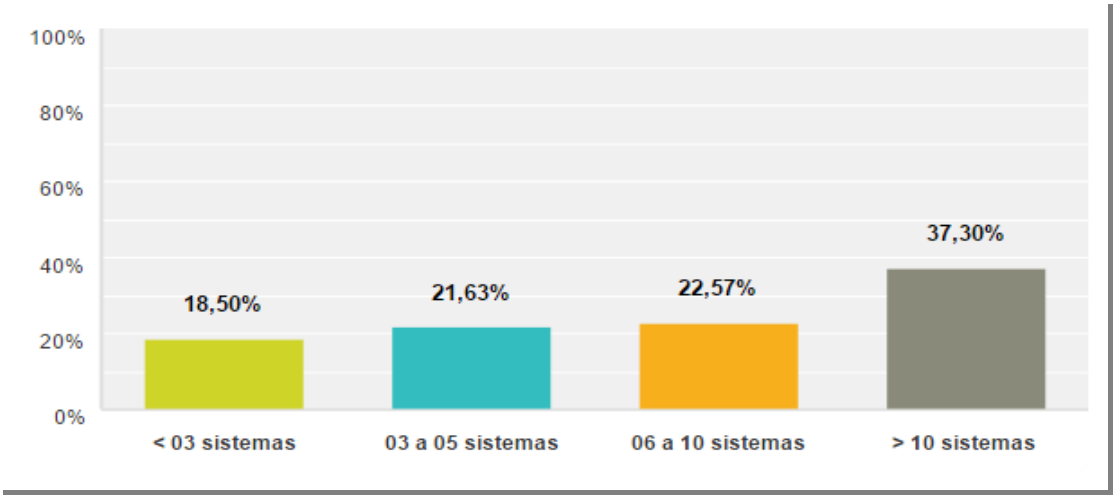

Gráfico 6 – Quantidade de sistemas desenvolvidos ou executados manutenções.

Em termos de quantidades de sistemas desenvolvidos ou executados atividades de manutenção, a maioria (37,30%) responderam que já desenvolveu ou executou mais de 10 sistemas, enquanto que os demais confirmaram (22,57%) entre 6 a 10 sistemas, seguindo de (21,63%) de 3 a 5 sistemas e para finalizar (18,50%) menos de 3 sistemas desenvolvidos ou executados. O Gráfico 6 mostra os resultados da pergunta sobre a quantidade de sistemas desenvolvidos para manutenção e percebe-se que quase 60% dos que responderam o questionário já havia desenvolvido mais de 06 sistemas em seus ambientes de trabalho.

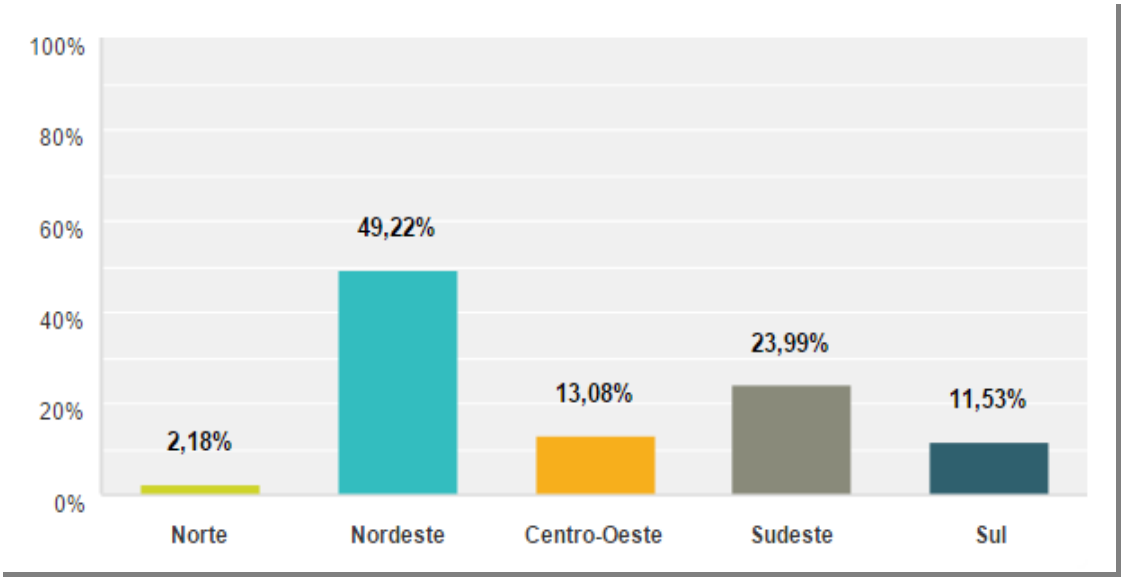

Gráfico 7 - Região do Brasil onde trabalha atualmente.

A pergunta do gráfico 7, investiga a região do Brasil que o entrevistado trabalha. As respostas obtidas mostram que a maioria dos respondentes são 49,22% do Nordeste, 23,99% Sudeste, 13,08% Centro-Oeste, 11,53% Sul e apenas 2,18% Norte. Nota-se que conseguimos mais respostas da região Nordeste, pelo fato dos remetentes possuíram mais contatos nessa região.

4.2 Análise das Questões da Pesquisa

A seções posteriores serão apresentados os resultados brutos e comparativos das 5 questões de pesquisa que foram criadas para atender os objetivos do survey:

4.2.1 Análise da Questão de pesquisa 01: Alguma técnica manual ou automática é utilizada para análise do código?

Para responder a essa primeira questão de pesquisa foi elaborada apenas uma pergunta com o objetivo de obter informações sobre a utilização de técnicas manuais e automáticas de análise de código.

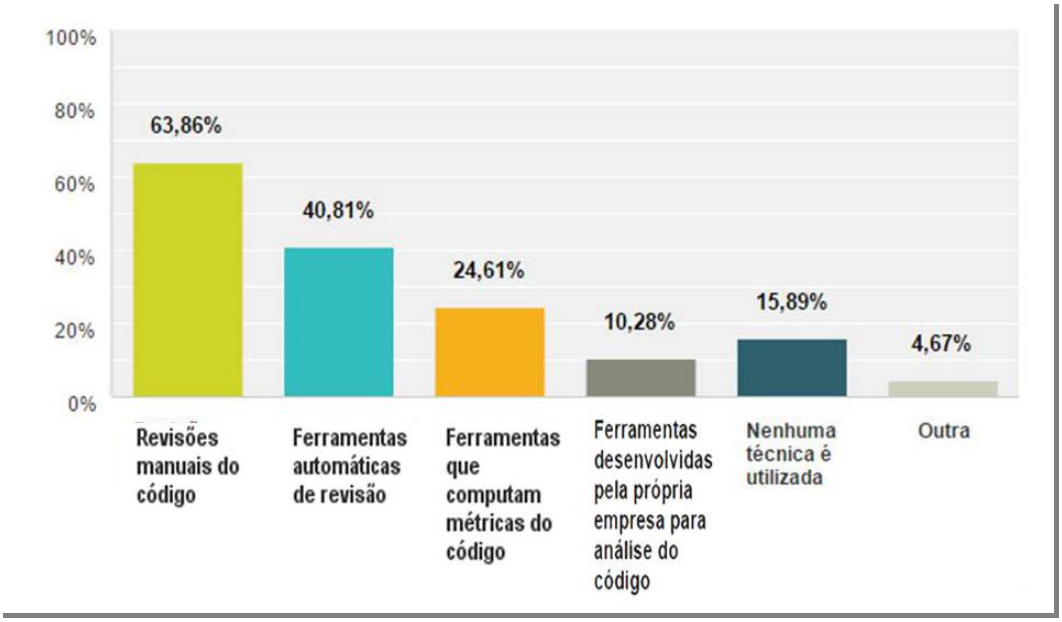

Grafico 8 – Técnicas utilizadas para analisar a qualidade do código nas empresas.

O Gráfico 8 apresenta os resultados dessa questão de múltipla escolha, mostrando que 63,86% dos entrevistados afirmaram fazer revisões manuais no código. Seguindo de 40,81% utilizam ferramentas automáticas de revisão, 24,61% dos respondentes afirmaram utilizar ferramentas que computam métricas e apenas 15,89% respondeu que nenhuma técnica utilizada. Outros 10,28% informaram que utilizam ferramentas desenvolvidas pela própria empresa e 4,67% selecionaram que utilizam outras técnicas. O resultado mostra que apesar das facilidades demonstradas pelas ferramentas de análise automática de código menos de 41% dos entrevistados afirmou utilizá-las. O mesmo acontece com a análise de código através de métricas onde menos de 25% afirmaram utilizar algum tipo de medição.

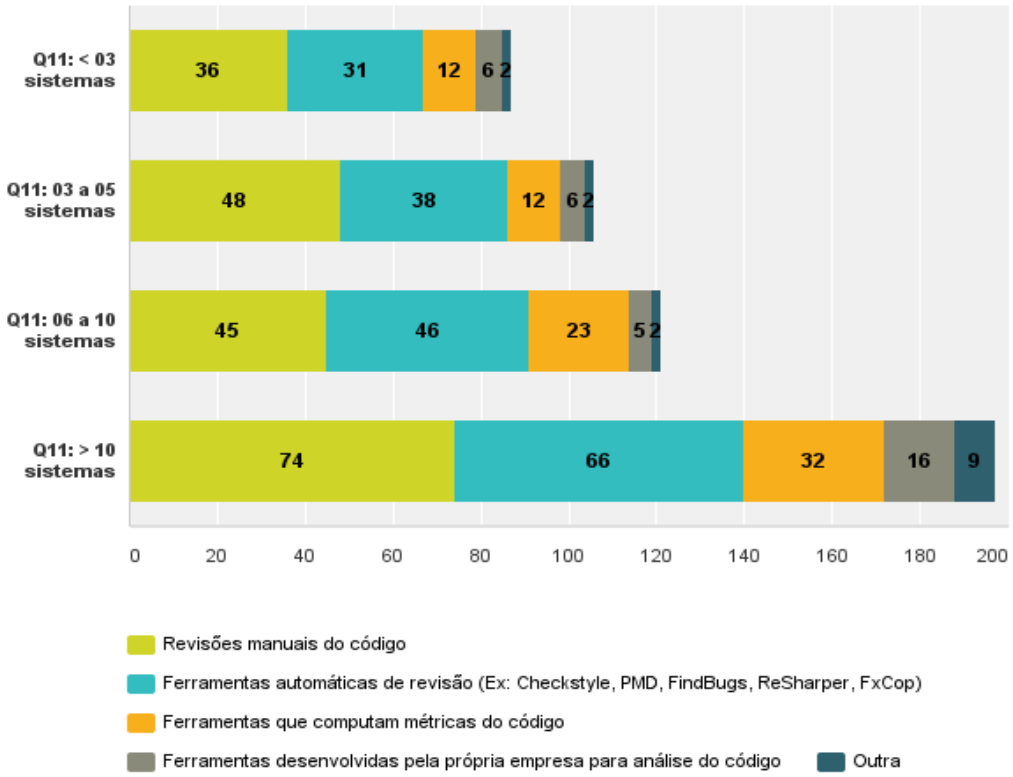

Gráfico 9 – Resultado Comparativo: Técnica X quantidade de sistemas desenvolvidos

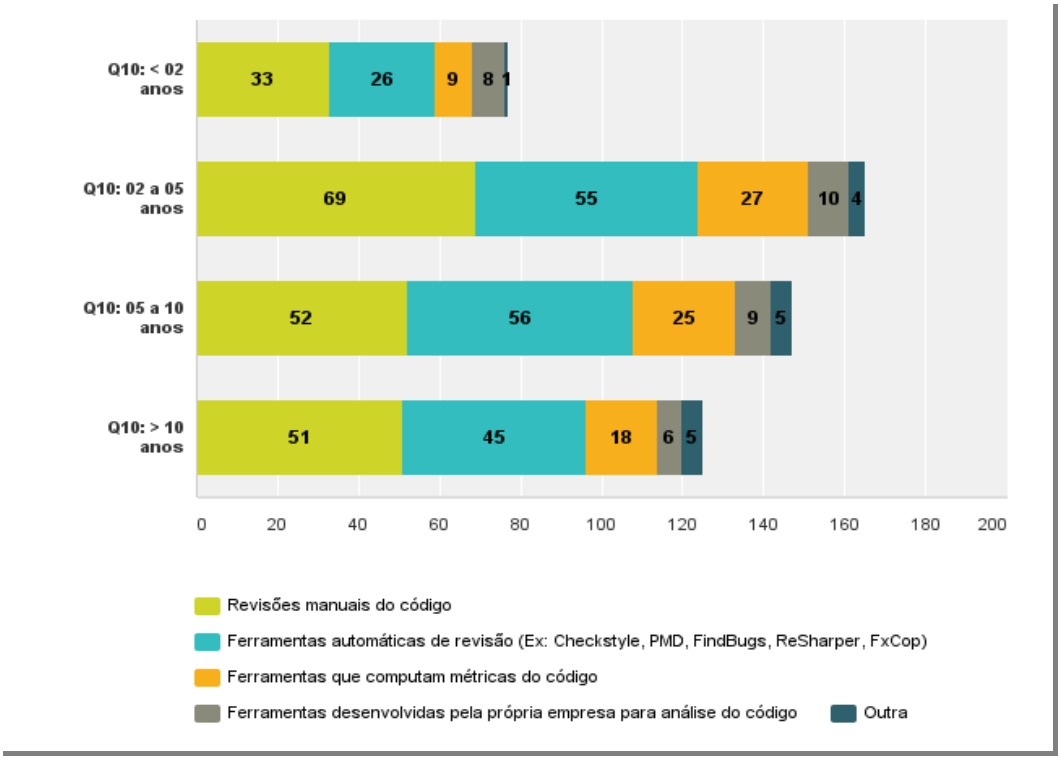

Gráfico 10– Resultado Comparativo: Técnica X tempo de experiência na área

O Gráfico 9 representa os resultados comparativo da técnica utilizada com a o tempo de quantidade de sistemas desenvolvidos. Nota-se que quanto maior o número de sistemas

desenvolvido (>10), maior é a utilização de técnicas para avaliação de qualidade. Seguindo essa mesma análise, o Gráfico 10 resultou que desenvolvedores entre 02 a 10 anos de experiência na área, além de realizarem revisões manuais de código também, também possui boa aceitação em relação as ferramentas de análise automática de código. Mostrando maior difusão dessas ferramentas entre os desenvolvedores mais experientes (45%).

Em resumo, chegamos as seguintes conclusões em relação à primeira questão de pesquisa:

a) Mais que 85% usam algum tipo de técnica;
b) A principal técnica utilizada é a de revisão manual (63,86%);
c) A utilização das técnicas é mais utilizada por profissionais com mais tempo de experiência na área.

4.2.2 Análise da Questão de pesquisa 02: Quais os níveis de importância dado para utilização das técnicas?

Para responder a segunda pergunta de pesquisa foram elaboradas duas perguntas que comparam o nível de importância dado pelo desenvolvedor e pela empresa.

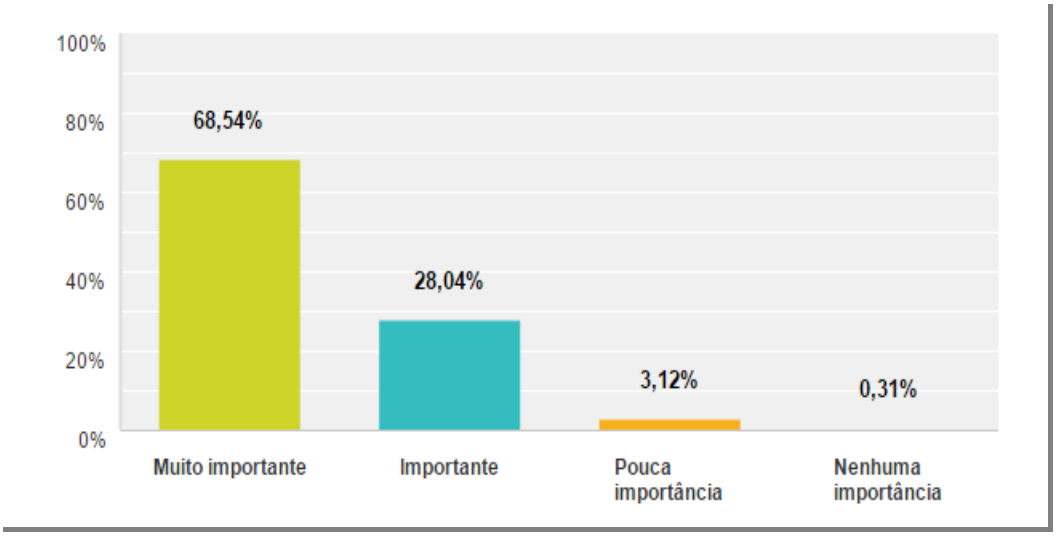

Gráfico 11 - Nível de importância atribuído por você para manutenção do código.

O Gráfico 11 exibe os resultado em relação ao nível atribuído ao desenvolvedor, a maioria dos entrevistados (68,54%) consideram o nível de importância para manutenção muito importante, enquanto (28,04%) importante, (3,12%) pouca importância e (0,31%) considera nenhuma importância. Percebe-se que mais de 96% dos respondentes consideram importante manter a qualidade do código desenvolvido.

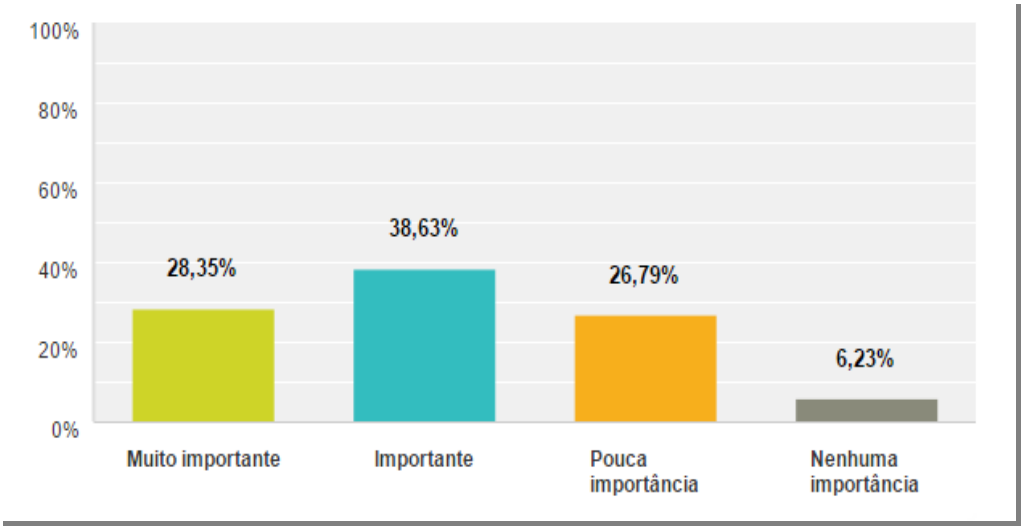

Gráfico 12 - Nível de importância atribuída pela empresa

Em termos de nível de importância atribuída pela empresa que o respondente trabalha (gráfico 12), a maioria (38,63%) respondeu que a empresa em questão considera importante a manutenção de qualidade do código. Por sua vez, (26,79%) acha a manutenção pouca importante, seguida de (28,35%) que consideram muito importante e apenas (6,23%) responderam que a empresa não atribui nenhum tipo de importância nesse assunto.

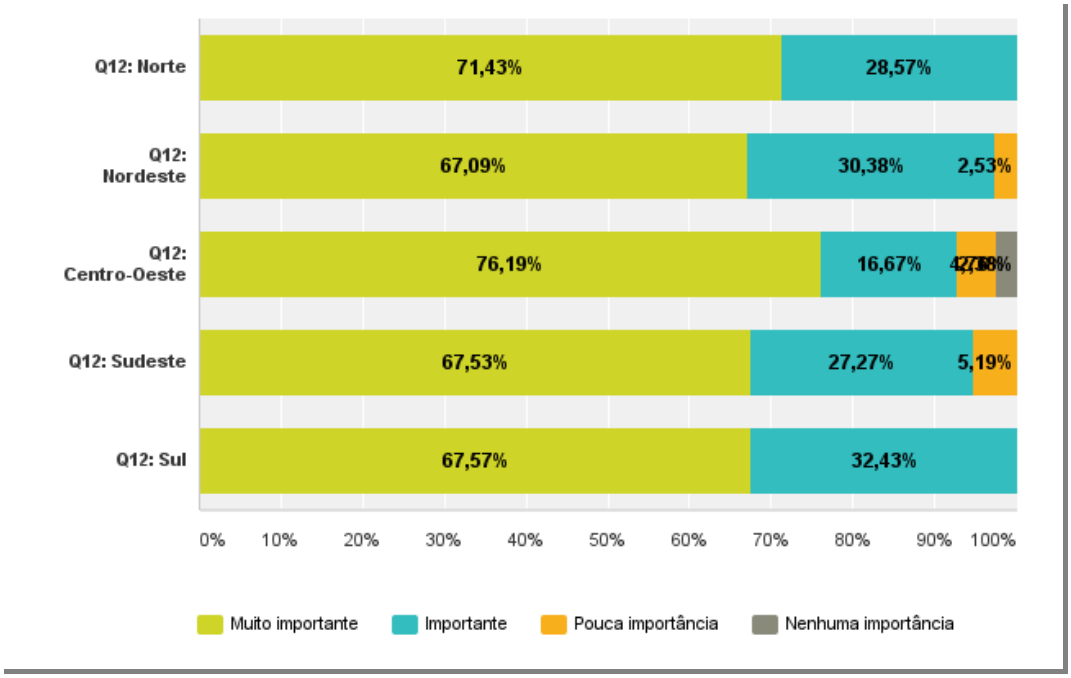

Gráfico 13 – Resultado Comparativo: Nível de importância X Região do Brasil

A respeito dos resultados da importância atribuída ao desenvolvedor (gráfico 13), observou-se que a maioria dos respondentes da região Centro-Oeste foi o que mais se destacou em relação a considerar a manutenção da qualidade do código como sendo muito importante (76,19%).

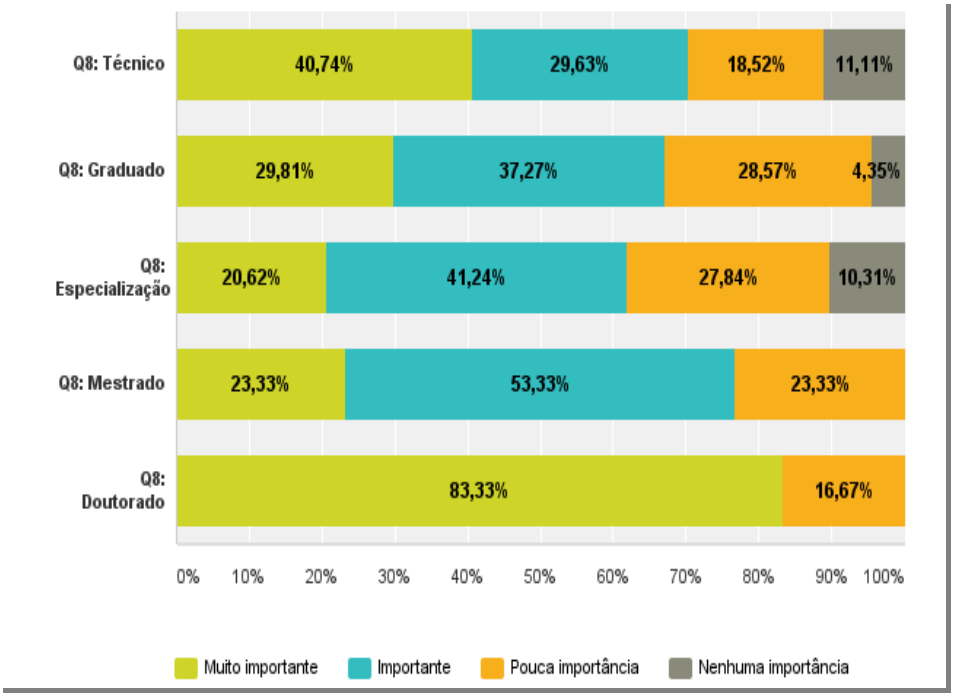

Gráfico 14 – Resultado Comparativo: Nível de importância da empresa X Formação

O Gráfico 14 apresenta em termos de níveis de importância atribuída pela empresa que o respondente trabalha e a sua formação. Os doutores demonstraram que as empresas onde os menos trabalham ou trabalhavam possuía um diferencial em relação a manutenção de qualidade, consideravam esse tema muito importante.

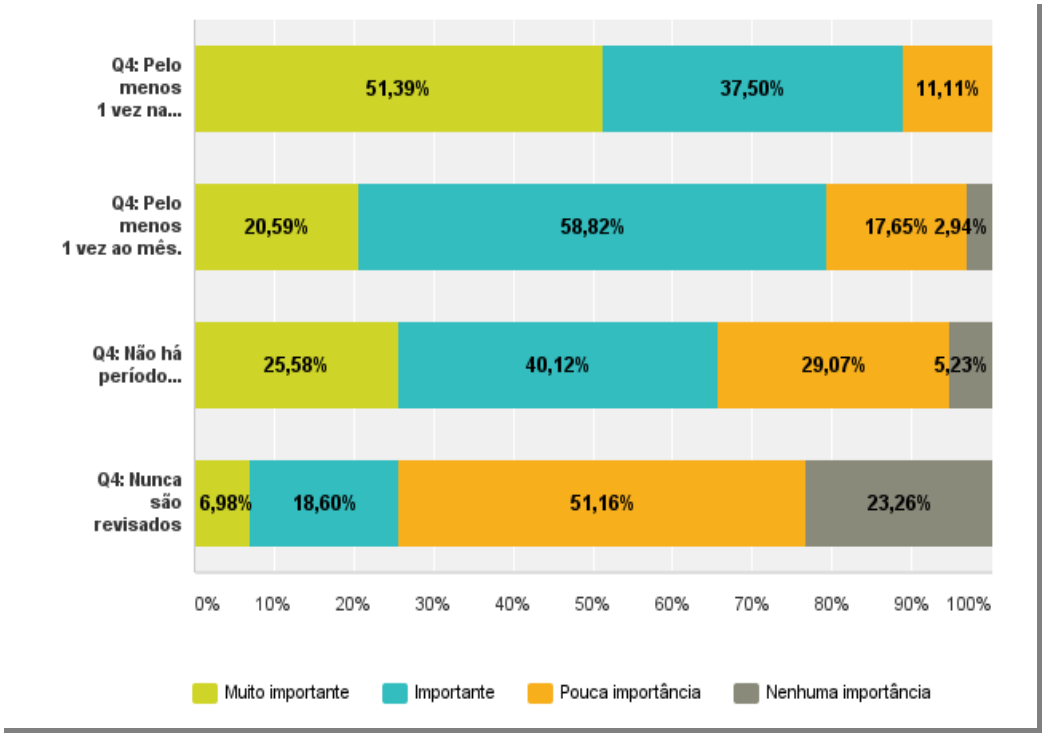

Gráfico 15 – Resultado Comparativo: Nível de importância da empresa X momento de recomendar qualidade

Seguindo a mesma questão, foi analisada no gráfico 15 que as empresas que atribuem níveis altos de importância na manutenção, revisam seus códigos uma vez por semana com taxa de percentual de 88,89%.

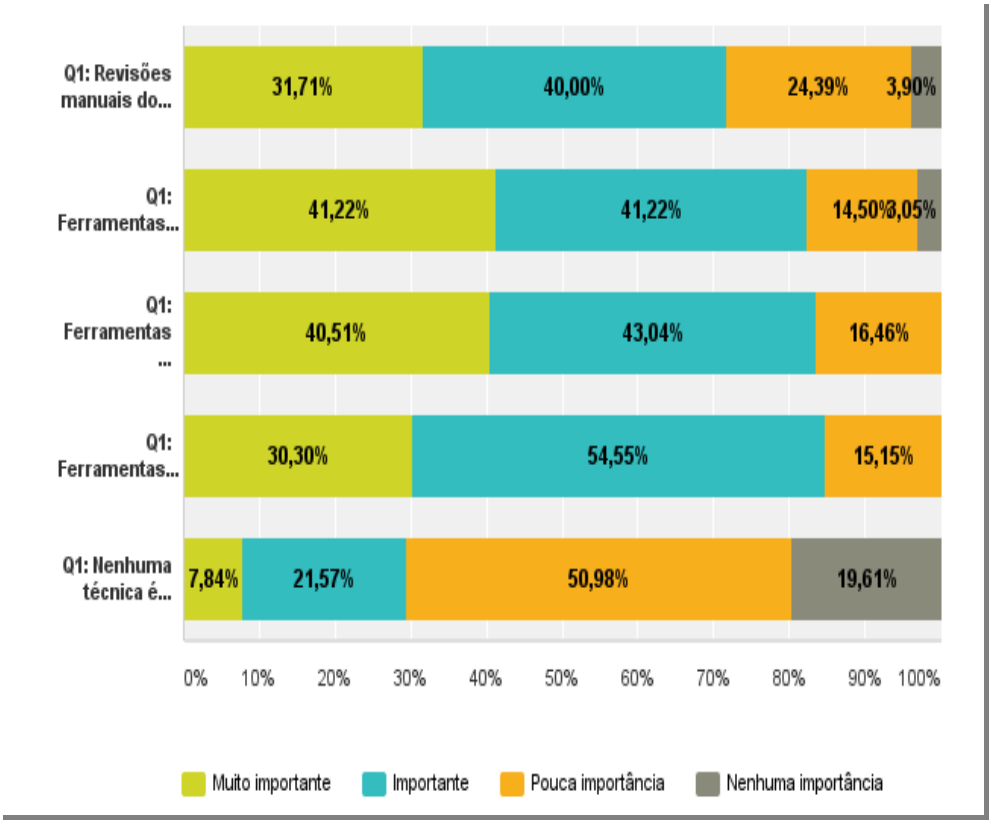

Gráfico 16 – Resultado Comparativo: Nível de importância da empresa X Técnicas

Em relação as ferramentas desenvolvidas pela própria instituições, como visto no gráfico 16. 54,55% dos respondentes que avaliaram como importante, utilizam ferramentas da própria empresa.

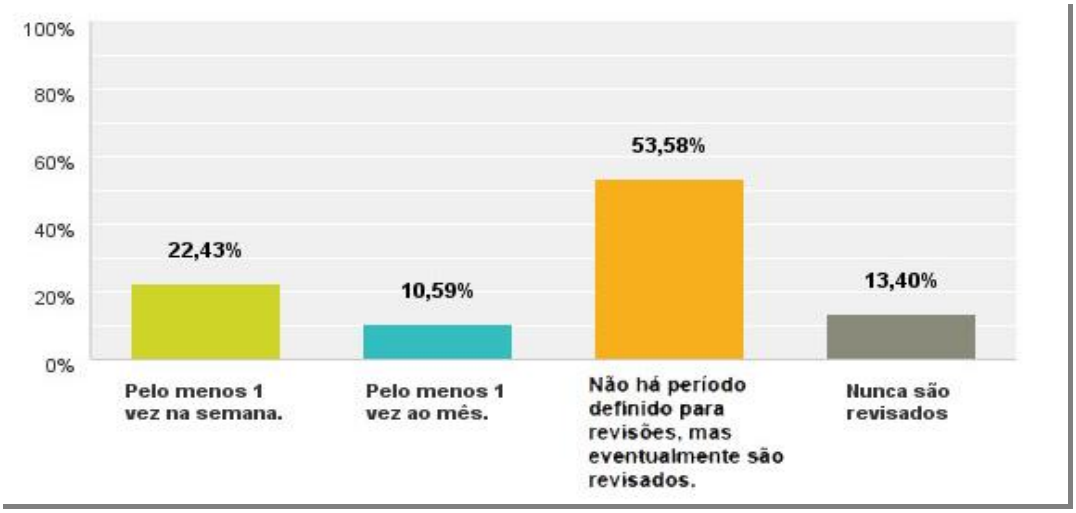

Gráfico 17 – Periodicidade de revisões da qualidade do código.

No respeitante à pergunta da periodicidade de revisão (Gráfico 17), a maioria dos entrevistados (53,58%) responderam que não há período definido para revisões, mas que eventualmente são revisados. Outras funções com um significante número de respostas foram: (22,43%) pelo menos uma vez na semana faz periodicidade de revisões no código, (13,40%) nunca faz revisão, e (10,59%) faz pelo menos 1 vez ao mês.

Em resumo, chegamos as seguintes conclusões em relação à segunda questão de pesquisa:

a) 66,98% dos respondentes consideram importante o uso de técnicas para análise do código;
b) Quanto maior o grau de formação, mais é considerado como importante as técnicas;
c) As empresas que consideram as técnicas muito importantes, a maioria disponibilizam em sua empresa suas próprias ferramentas (54,58%);
d) 53,58% dos respondes não dão muito importância a periodicidade da revisão do código.

4.2.3 Análise da Questão de pesquisa 03: Quais as dificuldades para utilização das técnicas automáticas?

Para responder a questão de pesquisa 03 foi criada uma pergunta cujos resultados são discutidos abaixo:

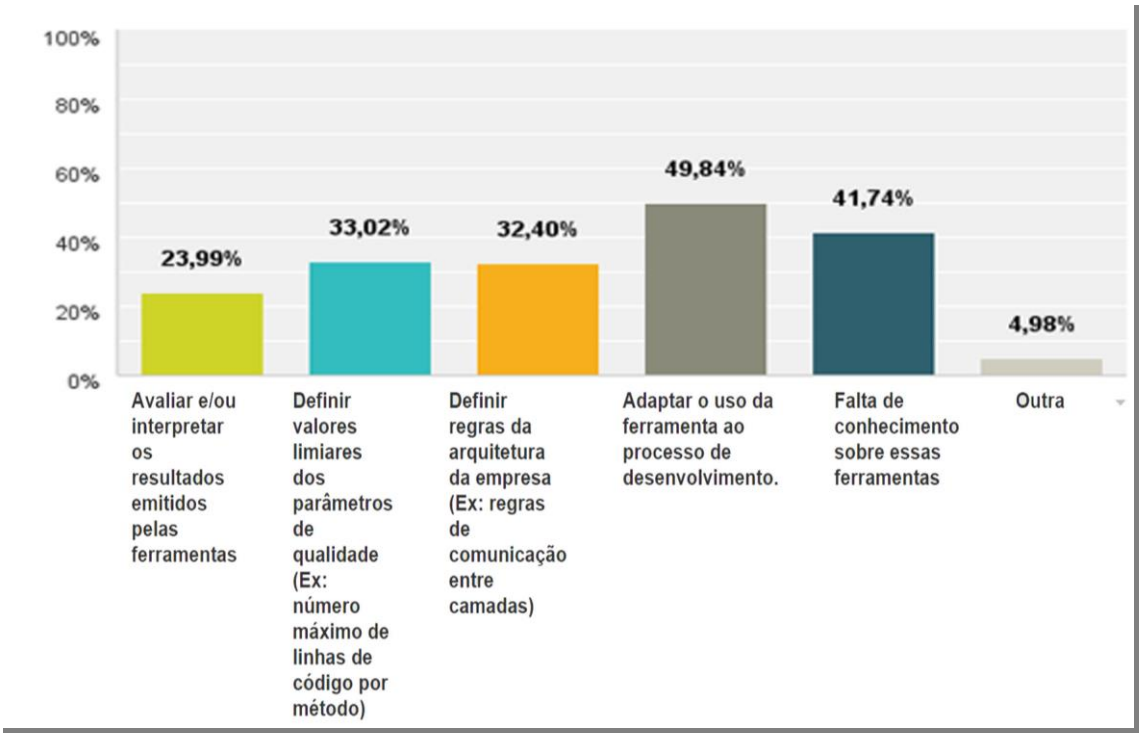

Gráfico 18–Principais dificuldades na utilização de ferramentas automáticas.

A pergunta refere-se às dificuldades na utilização de ferramentas automáticas para análise do código-fonte. O Gráfico 18 apresenta as respostas obtidas na pergunta 5 (pergunta de múltipla escolha). A maior parte dos respondentes (49,84%) informou dificuldades na fase de adaptação da ferramenta ao processo de desenvolvimento. Mas um número considerável (41,74%) também informou ter dificuldade pela falta de conhecimento sobre as ferramentas, e (32,40%) afirmou difícil definir valores aos parâmetros de qualidade, enquanto as outras dificuldades apresentadas foram: (33,02%) em definir regras da arquitetura na empresa, (23,99%) em avaliar os resultados obtidos pela ferramenta e (4,98%) responderam outras. Mostrando assim através desses resultados, a necessidade de divulgação ou incentivo das empresas no uso dessas técnicas.

Em resumo, chegamos a seguinte conclusão em relação à terceira questão de pesquisa: Que as principais dificuldades de utilização da ferramenta, são: a adaptação do uso da ferramenta e a falta de conhecimento.

4.2.4 Análise da Questão de pesquisa 04: O contexto da classe dentro da arquitetura deve ser considerado pelas técnicas de análise?

Para responder a quarta questão de pesquisa também foi elaborada apenas uma pergunta. Nos testes-pilotos realizados no questionário essa foi a questão apresentada com maior dificuldade.

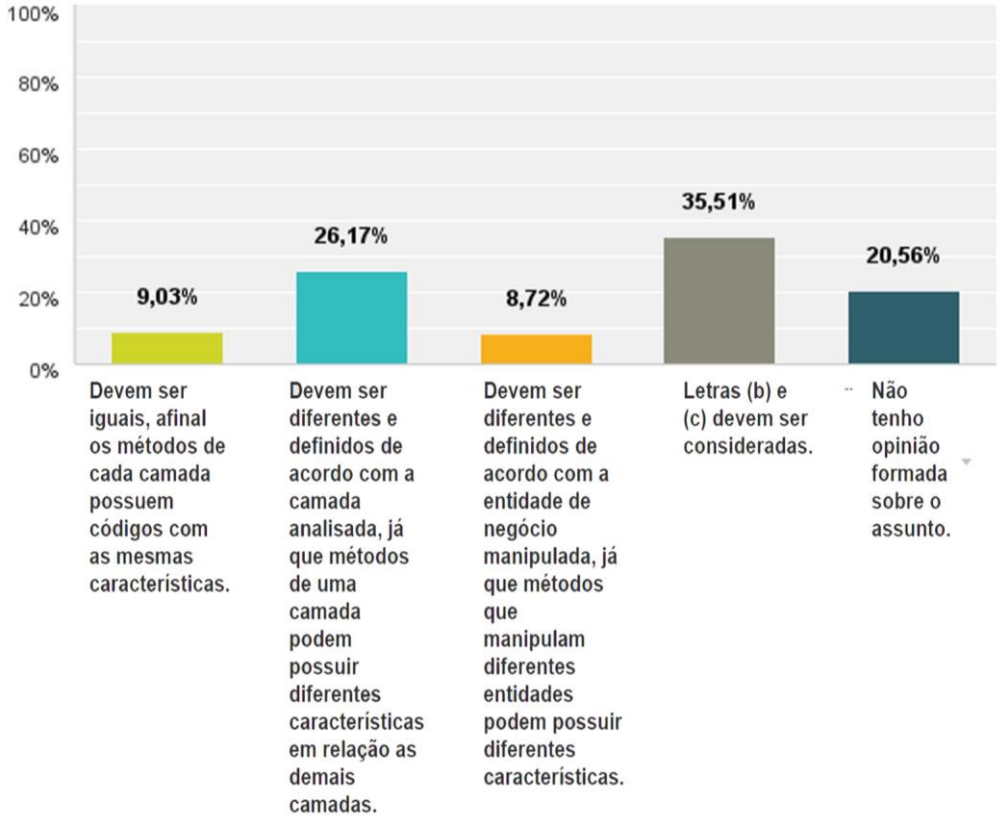

Gráfico 19 – Opinião em relação aos valores de parâmetros para camada.

O gráfico 19 mostra a opinião dos desenvolvedores a respeito dos valores atribuídos nas ferramentas nos parâmetros de camadas. As respostas foram: (35,51%) consideram as letras b e c como validas. (26,17%) responderam que devem ser diferentes de acordo com a camada analisada e (8,72%) que devem ser diferentes de acordo com a entidade de negócio. Seguindo de (20,56%) os profissionais não possuem opinião formada sobre esse assunto. Portanto a maioria dos respondetes, deixou claro que o valores devem ser diferentes para cada camada. Esses resultados vão de encontro as ferramentas que analisam métricas de código e definem valores limiares para análise que são considerados para todo sistema. Dos respondentes apenas 9,03% consideram que os valores limiares devem ser únicos e mais de 70% dos respondentes consideraram que esses valores precisam ser ajustados de acordo com o componente da arquitetura analisado e/ou entidade de negócio manipulada.

Uma análise interessante dessa questão foi na caracteristica encontrada dos respondentes. Desenvolvedores que não possuem uma opniao formada no assunto de ferramentas que ajustam paramêtros (ou no aspecto de conhecimento ou por não utilizar em sua empresa), apresentaram as seguintes análises:

- 37,88% está entre 2 a 5 anos de experiência na área; (gráfico 20)

- 35 desenvolvedores responderam que a dificuldade maior de utilizaçao da ferramenta foi a falta de connhecimento. (gráfico 21)

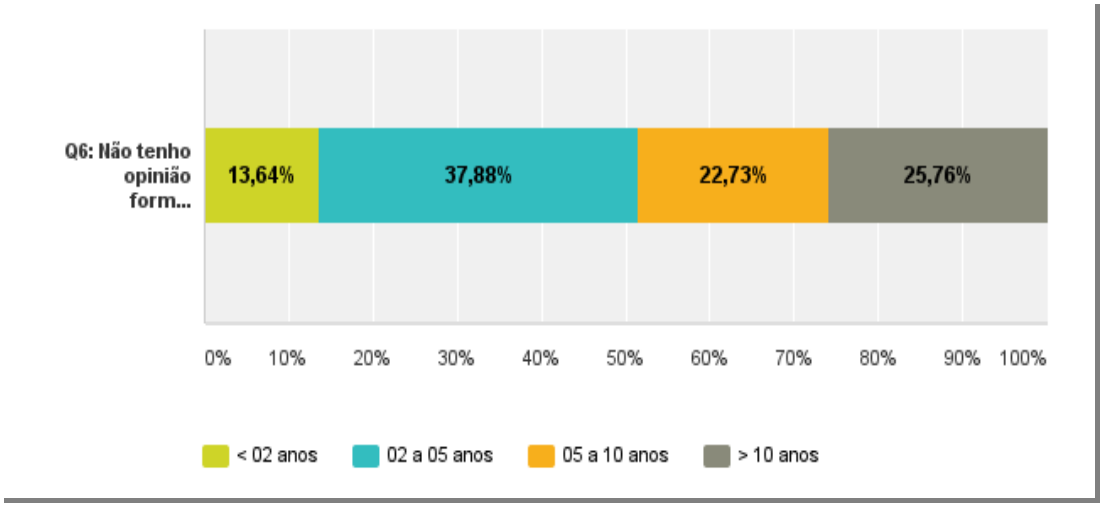

Gráfico 20 – Resultado Comparativo: Opinião em relação aos valores de parâmetros X Tempo de experiência

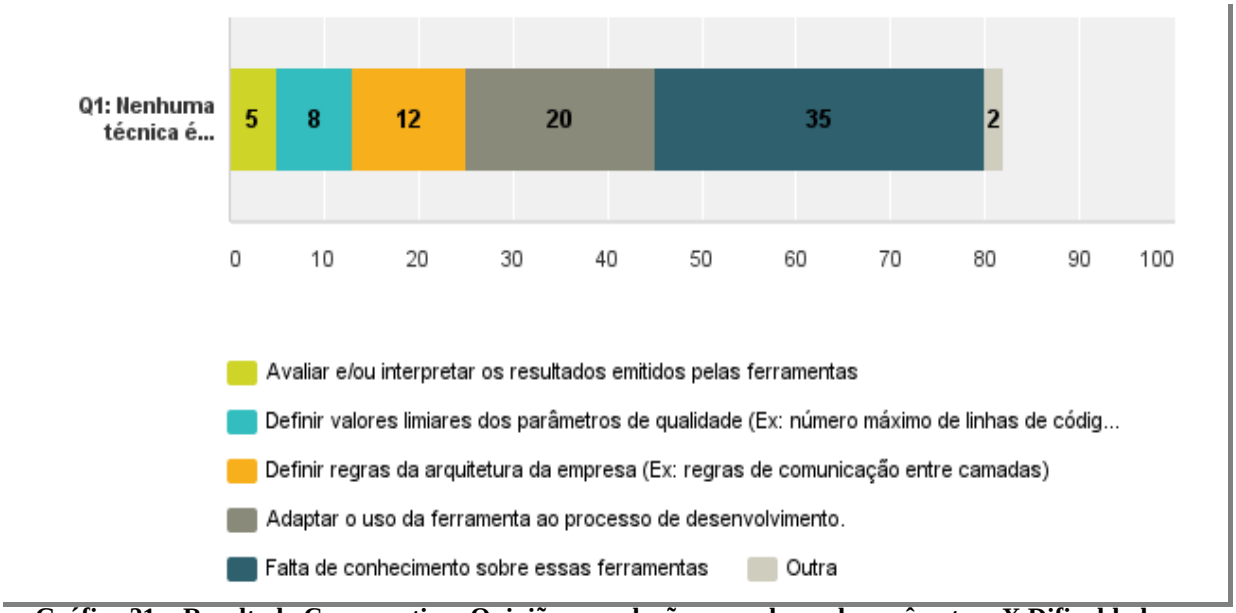

Gráfico 21 – Resultado Comparativo: Opinião em relação aos valores de parâmetros X Dificuldades

Em resumo, chegamos as seguintes conclusões em relação à quarta questão de pesquisa:

a) 35,51% responderam que devem ser consideradas desde que sejam diferentes e definidos de acordo com a camada e a entidade de negócio;

b) 51,52% dos respondentes que não entendem o assunto, possui pouca experiência na área.

4.2.5 Análise da Questão de pesquisa 05: Qual o melhor momento para aplicação das técnicas automáticas?

Para última questão de pesquisa foi criada apenas uma questão de escolha exclusiva, cujos dados são discutidos a seguir.

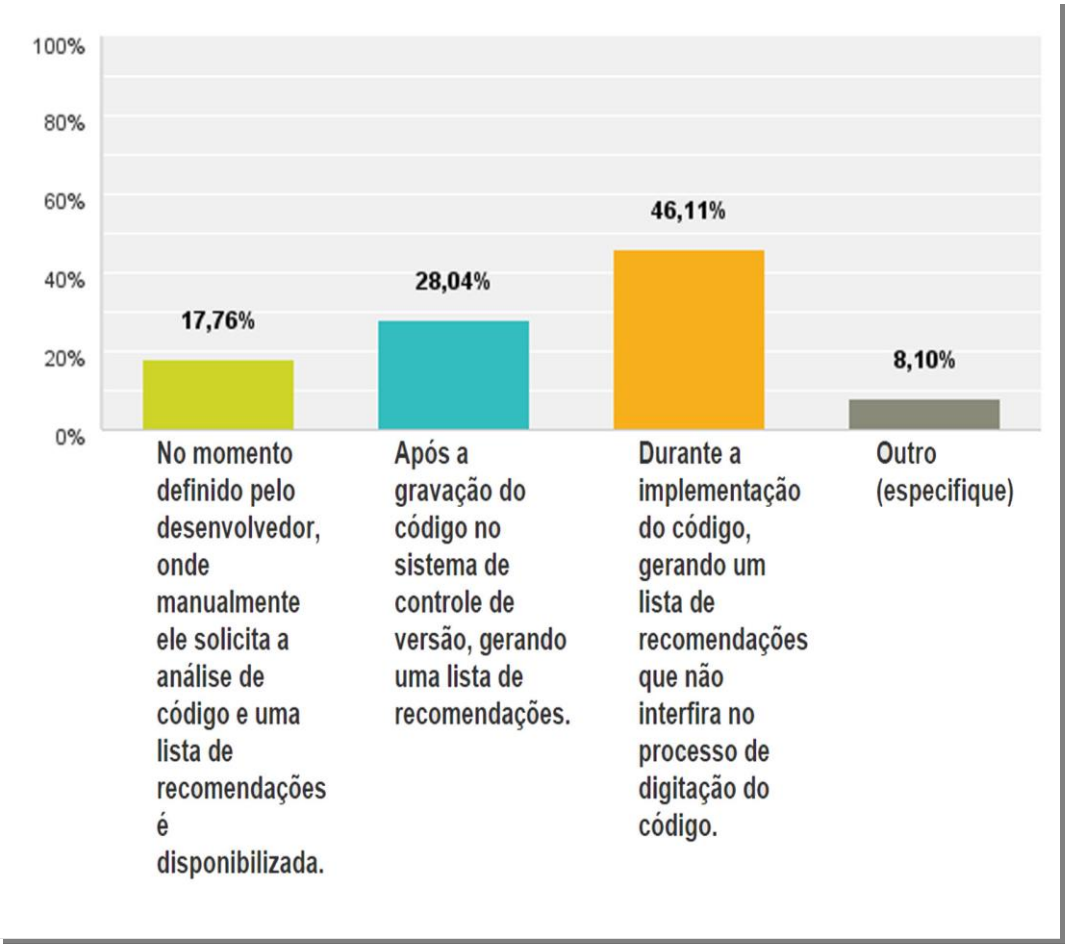

Gráfico 22 – Melhor momento e formato para recomendar aos desenvolvedores possíveis problemas de qualidade no código.

Existem diferentes opiniões a respeito do melhor momento de recomendar ao desenvolvedor possíveis problemas de qualidade no código, apresentadas no Gráfico 22. Os resultados mostram que 46,11% responderam que o melhor momento é durante a implementação do código, gerando uma lista de recomendações que não interfira no processo, (28,04%) confirmam após a gravação do código no sistema de controle de versão. Por sua vez, (17,76%) tem preferência no momento definido pelo desenvolvedor, onde manualmente ele solicita a análise de código. E por fim, (8,10%) responderam outros. Observou-se que os respondentes preferem a análise durante a implementação, desde que não atrapalhe no andamento do código.

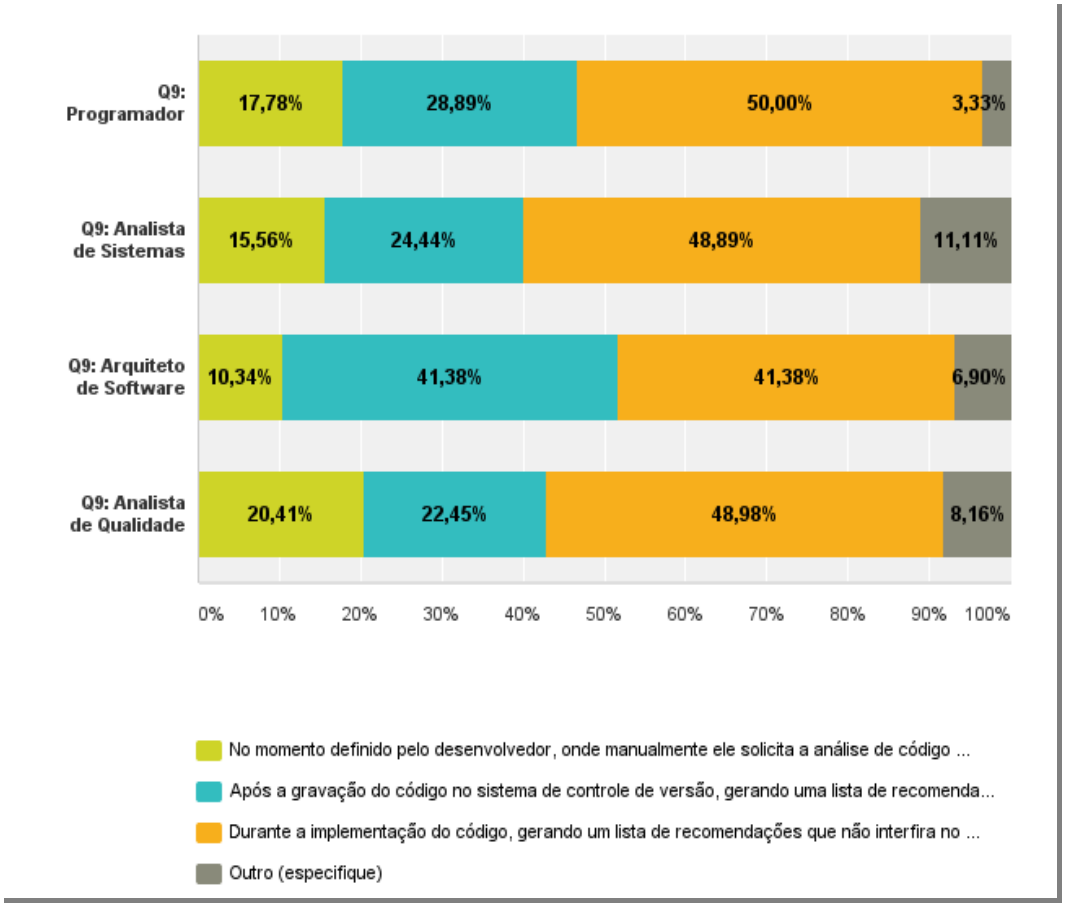

Gráfico 23– Resultado comparativo: Principais dificuldades na utilização de ferramentas automáticas X posição na empresa

A pesquisa no resultante comparativo dessa questão (gráfico 23), obteve como resultado diferencial a seguinte análise: Todos os profissionais preferem que a recomendação seja no momento da implementação, mas os arquitetos de software tiveram divergências de respostas. Os mesmos atribuíram (41,38%) na fase de implementação e após a gravação do código. Esse dado pode ser justificado porque a questão era de escolha exclusiva e os arquitetos preocupam-se principalmente com o código gravado no repositório.

Em resumo, chegamos as seguintes conclusões em relação à cinco questão de pesquisa:

a) Que o melhor momento de recomentar ao desenvolvedor é durante a implementação do código (46,11%);
b) 82,76% dos respondentes que são arquitetos de software preferem a recomendação na implementação e após a gravação do código.

5 CONCLUSÃO

Após realizarmos a pesquisa e analisarmos os dados, foi possível concluir que o documento atingiu o objetivo proposto de caracterizar as técnicas que evitam erosão arquitetural, delimitados aos desenvolvedores das empresas de TI do Brasil. Além de atender a necessidade de uma pesquisa com esse tema, já que não existia nenhum tipo especifico realizado anteriormente. Através da pesquisa foi possível mensurar o nível de importância tanto dos desenvolvedores, quanto das empresas no uso dessas técnicas.

Para atender o objetivo, realizou-se uma pesquisa survey, que foi aplicada utilizando uma ferramenta que disponibilizou o questionário na Web. A pesquisa foi acessada por 375 desenvolvedores, deste 85,6% responderam por completo o questionário. O questionário possuía 12 perguntas distribuídas entre perguntas sobre o perfil e especificas para responder a cinco questões de pesquisas definidas no planejamento.

A pesquisa apontou diferentes conclusões, entre elas:

(1) Mais de 85% dos entrevistados conhecem ou usam algum tipo de técnica que analisa a qualidade do código-fonte (manuais ou/e automáticas);

(2) Mais de 90% responderam que as principais dificuldades encontradas são a adaptação do uso da ferramenta ao processo de desenvolvimento e a falta de conhecimento sobre as ferramentas;

(3) Mais de 80% dos respondentes com perfil de desenvolvedores que possuem maior nível de formação, consideram as técnicas importantes;

(4) 70% dos respondentes consideraram que valores limiares das métricas precisam ser ajustados de acordo com o componente da arquitetura analisado e/ou entidade de negócio manipulada;

(5) 46,11% responderam que o melhor momento de recomendar problemas no código é durante a implementação.

Através da conclusão destes dados, pesquisas futuras podem ser úteis para criar ou aprimorar novas técnicas. Empresas poderão observar a necessidade de instalação de ferramentas automáticas e futuros investimentos podem ser realizados com técnicas que são mais aceitas no mercado.

6 REFERENCIAS BIBLIOGRÁFICAS

ABREU, F. B. As Métricas na Gestão de Projectos de Desenvolvimento de Sistemas de Informação. **6as Jornadas para a Qualidade no Software APQ Lisboa**, p. 1–16, 1992. Disponívelem:<http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:As+M?tricas+na+Gest?o+de+Projectos+de+Desenvolvimento+de+Sistemas+de+Informa??o#0>. .

AGNER, L.T.W., SOARES, I.W., STADZISZ, P.C., SIMÃO, J.M.: A brazilian survey on UML and model-driven practices for embedded software development. J**ournal of Systems and Software**. Disponível em: http://www.sciencedirect.com/science/article/pii/S0164121212003160

ALDRICH, J.; CHAMBERS, C.; NOTKIN, D. Architectural reasoning in ArchJava. **Ecoop 2002 - Object-Oriented Programming**, v. 2374, p. 334–367, 2002.

ARCOVERDE, R.; GARCIA, A.; FIGUEIREDO, E. Understanding the longevity of code smells. Workshop on Refactoring tools (WRT). **Anais...** p.33–36, 2011. Disponível em: <http://portal.acm.org/citation.cfm?doid=1984732.1984740>. .

BABBIE, Earl. Methods of Survey Research. 1999

BASS, L.; CLEMENTS, P.; KAZMAN, R. **Software Architecture in Practice**. 2003.

BERTRAN, I. M. Detecting architecturally-relevant code smells in evolving software systems. **2011 33rd International Conference on Software Engineering (ICSE)**, p. 1090–1093, 2011.

CHECKSTYLE. Disponível em: <**http://checkstyle.sourceforge.net/**Acesso em: 14 fev. 2015

CHESS, B.; MCGRAW, G. Static analysis for security. **Security \& Privacy, IEEE**, p. 76–79, 2004.

CLEMENTS, P.; GARLAN, D.; LITTLE, R.; NORD, R.; STAFFORD, J. Documenting software architectures: views and beyond. **25th International Conference on Software Engineering, 2003. Proceedings.**, 2003.

COUPER, M. Web surveys: a review of issues and approaches. **Public opinion quarterly**, v. 64, p. 464–494, 2000.

FINK, A. How to sample in surveys. Thousand Oaks: Sage, 1995. (The Survey Kit, v.6).

FONTANA, F. A.; MARIANI, E.; MORNIROLI, A.; SORMANI, R.; TONELLO, A. An experience report on using code smells detection tools. Proceedings - 4th IEEE International Conference on Software Testing, Verification, and Validation Workshops, ICSTW 2011. **Anais...** p.450–457, 2011.

FOWLER, M.; BECK, K.; BRANT, J.; OPDYKE, W.; ROBERTS, D. Refactoring: Improving the Design of Existing Code. **Xtemp01**, p. 1–337, 1999.

FREITAS, H.; OILIVEIRA, M.; SACCOL, A. Z.; MOSCAROLA, J. O método de pesquisa survey. **Revista de Administração**, p. 105–112, 2000.

GARLAN, D.; SHAW, M. An Introduction to Software Architecture. **Knowledge Creation Diffusion Utilization**, v. 1, p. 1–40, 1994. Disponível em: <http://portal.acm.org/citation.cfm?id=865128>. .

GIL, A. C. **Métodos e técnicas de pesquisa social**. 1999.

HOLT, J. Current practice in software engineering : , , n. November 1995, 1997.

KALIBROBR. Disponível em: <**http://ccsl.ime.usp.br/kalibro**/ Acesso em: 15 fev. 2015

KASUNIC, M. *Design na Efective Survey; 2005.*

MEDVIDOVIC, N.; TAYLOR, R. N. A classification and comparison framework for software architecture description languages. **IEEE Transactions on Software Engineering**, v. 26, 2000.

PARNAS, D. L. Software aging. **Proceedings of 16th International Conference on Software Engineering**, 1994.

PASSOS, L.; TERRA, R.; VALENTE, M. T.; DINIZ, R.; CHAGAS MENDONÇA, N. DAS. Static architecture-conformance checking: An illustrative overview. **IEEE Software**, p. 82–89, 2010.

PUNTER, T.; CIOLKOWSKI, M.; FREIMUT, B.; JOHN, I. Conducting on-line surveys in software engineering. **2003 International Symposium on Empirical Software Engineering, 2003. ISESE 2003. Proceedings.**, 2003.

QUESTIONFORM. Disponível em: <http://questionform.com/> Acesso em: 14 fev. 2015

ROST, D.; NAAB, M.; LIMA, C.; FLACH GARCIA CHAVEZ, C. VON. Software architecture documentation for developers: A survey. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). **Anais...** v. 7957 LNCS, p.72–88, 2013.

SAUER, C.; ROSS JEFFERY, D.; LAND, L.; YETTON, P. The effectiveness of software development technical reviews: A behaviorally motivated program of research. **IEEE Transactions on Software Engineering**, p. 1–14, 2000.

SILVA, L. DE; BALASUBRAMANIAM, D. Controlling software architecture erosion: A survey. **Journal of Systems and Software**, v. 85, p. 132–151, 2012.

SLINGER, S. Code Smell Detection in Eclipse. **Science**, p. 1–69, 2005.

SOMMERVILLE, I. **Software engineering**. 2011.

SONAR. Disponível em: <http://www.sonarsource.org/downloads/ Acesso em: 14 fev. 2015

SURVEYMONKEY. Disponível em: <http://www.surveymonkey.com> Acesso em: 12 fev. 2015

TSANTALIS, N.; CHAIKALIS, T.; CHATZIGEORGIOU, A. JDeodorant: Identification and removal of type-checking bad smells. Proceedings of the European Conference on Software Maintenance and Reengineering, CSMR. **Anais...** p.329–331, 2008.

APÊNDICE

Questionário
Quais técnicas você utiliza para avaliação da qualidade do código-fonte?

São apenas **12** perguntas que podem ser respondidas em aproximadamente 05 minutos.

Esta pesquisa tem como objetivo aprimorar técnicas usadas para avaliação da qualidade do código-fonte em desenvolvimento. Algumas características de um código com qualidade são:

• Obedecer as definições da arquitetura de software propostas pela empresa.
• Evitar métodos longos e classes muito grandes
• Evitar implementações complexas com várias estruturas de repetição ou condicionais encadeadas ou aninhadas.

Revisões manuais de artefatos desenvolvidos pela equipe, utilização de ferramentas automáticas de revisão (Ex: PMD, FindBugs, Checkstyle, ReSharper, FxCop) e análise de métricas coletadas no código são algumas das abordagens utilizadas.

Agradecemos a participação e se desejar receber os resultados da pesquisa favor informar o e-mail ao final do questionário.

1. Quais técnicas são utilizadas para analisar a qualidade do código desenvolvido pela empresa? Mais de uma opção pode ser selecionada.
( ) Revisões manuais do código
( ) Ferramentas automáticas de revisão (Ex: Checkstyle, PMD, FindBugs, ReSharper, FxCop)
( ) Ferramentas que computam métricas do código
( ) Ferramentas desenvolvidas pela própria empresa para análise do código
( ) Nenhuma técnica é utilizada
( ) Outra _____

2. Qual o nível de importância atribuído por <u>você</u> para manutenção da qualidade do código da aplicação?
( ) Muito importante
( ) Importante
( ) Pouca Importante
( ) Nenhuma importância

3. Qual o nível de importância atribuído pela <u>empresa</u> para manutenção da qualidade do código da aplicação?
( ) Muito importante
( ) Importante
( ) Pouca Importante
( ) Nenhuma importância

4. Qual a periodicidade de revisões da qualidade do código desenvolvido:
( ) Pelo menos 1 vez na semana

( ) Pelo menos 1 vez ao mês.
( ) Não há período definido para revisões, mas eventualmente são revisados
( ) Nunca são revisados

5. Quais as principais dificuldades na utilização de ferramentas automáticas para análise da qualidade do código-fonte? Mais de uma opção pode ser selecionada.

( ) Avaliar e/ou interpretar os resultados emitidos pelas ferramentas
( ) Definir valores limiares dos parâmetros de qualidade (Ex: número máximo de linhas de código por método)
( ) Definir regras da arquitetura da empresa (Ex: regras de comunicação entre camadas)
( ) Adaptar o uso da ferramenta ao processo de desenvolvimento.
( ) Falta de conhecimento sobre essas ferramentas
( ) Outra _____

6. As ferramentas permitem ajustar valores dos parâmetros utilizados para análise da qualidade do código, por exemplo, o número máximo de linhas em um método ou nível máximo de complexidade de um método. Considerando um sistema em três camadas (GUI, Negócio e Persistência) qual a sua opinião em relação aos valores desses parâmetros para cada camada?

( )Devem ser iguais, afinal os métodos de cada camada possuem códigos com as mesmas características.
( ) Devem ser diferentes e definidos de acordo com a camada analisada, já que métodos de uma camada podem possuir diferentes características em relação as demais camadas.
( ) Devem ser diferentes e definidos de acordo com a entidade de negócio manipulada, já que métodos que manipulam diferentes entidades podem possuir diferentes características.
( ) Letras (b) e (c) devem ser consideradas.
( ) Não tenho opinião formada sobre o assunto.

7. Na sua opinião, qual seria o melhor momento e formato para recomendar ao desenvolvedor possíveis problemas de qualidade no código-fonte desenvolvido?

( ) No momento definido pelo desenvolvedor, onde manualmente ele solicita a análise de código e uma lista de recomendações é disponibilizada.
( ) Após a gravação do código no sistema de controle de versão, gerando uma lista de recomendações.
( ) Durante a implementação do código, gerando um lista de recomendações que não interfira no processo de digitação do código.
( ) Outro (especifique) _____

8. Qual o seu nível de formação?

( ) Técnico
( ) Graduado
( ) Especialização
( ) Mestrado
( ) Doutorado

9. Qual a sua posição atual na empresa?

( ) Programador
( ) Analista de Sistemas
( ) Arquiteto de Software
( ) Analista de Qualidade

( ) Outra (especifique) _____

10. Quanto tempo de experiência possui na área de desenvolvimento?
( ) < 02 anos
( ) 02 a 05 anos
( ) 05 a 10 anos
( ) > 10 anos

11. Quantos sistemas já desenvolveu ou executou atividades de manutenção?
( ) < 03 sistemas
( ) 03 a 05 sistemas
( ) 06 a 10 sistemas
( ) > 10 sistemas

12. Região do Brasil onde trabalha atualmente?
( ) Norte
( ) Nordeste
( ) Centro-Oeste
( ) Sudeste
( ) Sul

13. Email (opcional): Resultados serão enviados para esse email
_____